AD-A050 206 UNIVERSITY OF SOUTH FLORIDA TAMPA F/G 9/2
THEORY AND COMPUTER PROGRAM FOR TRANSFER FUNCTION IDENTIFICATIO--ETC(U)
FEB 78 V K JAIN, G J DOBECK, L J LAWDERMILT N61331-75-C-0012

UNCLASSIFIED NCSL-TM-204-78 NL

1 OF 2

AD
A050 206

N61331-75-C-0012---Main Contract
N61339-75-C-0012---Subcontract No. 1-E-21A02
From Georgia Institute of Technology per
Ms. Marilyn Griffin, Naval Coastal Systems
Laboratory
per Telecon by J. Cundiff

DEPARTMENT OF THE NAVY
NAVAL COASTAL SYSTEMS LABORATORY
PANAMA CITY, FLORIDA 32407

IN REPLY REFER TO
Code 361:MG
22 February 1978

From: Commanding Officer
To: Distribution

Subj: NAVCOASTSYSLAB Technical Memorandum NCSL 204-78, "Theory and Computer Program for Transfer Function Identification by the GRAM Identifier with Application to Undersea Vehicles," prepared by University of South Flori[illegible] Feb 1978; errata to

Ref: (a) NCSL transmittal Code 116.6 of 1[illegible]b 1978

1. Reference (a) forwarded subject report to distribution.

2. Request all holders make the following errata pen-and-ink changes to the report on the pages indicated below:

a. Administrative Information (inside front cover) - Change Contract No. N61339-75-C-0122 to read: Subcontract No. 1-E-21A02 to Prime Contract N61339-75-C-0122 HR-02.

b. DD Form 1473:

(1) Change Report Number (block 1) to: NCSL TM-204-78

(2) Change Contract or Grant Number(s) (block 8) to:

N61331-75-C-0012
Subc. 1-E-21A02

(3) Change report date (block 12) to: February 1978

c. The report number at the top of all internal pages should be changed to NCSL TM-204-78.

F. B. LAWRENCE
By direction

Distribution
(see reverse)

INITIAL DISTRIBUTION LIST
NCSL REPORT TM 204-78

427	Commander, Naval Sea Systems Command	
	(SEA 032)	(Copy 1)
	(SEA 034)	(Copy 2)
	(SEA 035)	(Copy 3)
	(SEA 92)	(Copy 4)
	(PMS 393)	(Copy 5)
	(PMS 395)	(Copy 6)
	(Library)	(Copy 7)
1	Chief of Naval Material	
	(NMAT 08 T23 - Mr. Vittucci)	(Copy 8)
	Mr. Remson, Mr. Jaffe)	(Copies 9-10)
5	Commander, Naval Ship Engineering Center	
	(Library)	(Copy 11)
	(Code 6136 - Mr. Keane)	(Copy 12)
	(Code 6136 - Mr. Louis)	(Copy 13)
236	Commander, David Taylor Naval Ship R&D Center	
	(Library)	(Copy 14)
	(Code 1564 - Mr. Feldman)	(Copy 15)
	(Code 1548 - Mr. Folb)	(Copy 16)
266	Commanding Officer, Naval Underwater Systems Center	
	(Library)	(Copy 17)
	(Mr. Goodrich)	(Copy 18)
	(Mr. Nadolink)	(Copy 19)
265	Commander, Naval Oceans Systems Center	
	(Code 5332)	(Copy 20)
	(Code 6302)	(Copy 21)
	(Library)	(Copy 22)
484	Director, Naval Research Laboratory	
	(Library)	(Copy 23)
210	Commander, Naval Surface Weapons Center, White Oak	
	(Library)	(Copy 24)
463	Commander, Naval Surface Weapons Center, Dahlgren	
	(Library)	(Copy 25)
154	Superintendent, Naval Academy, Annapolis	
	(Library)	(Copy 26)
222	Superintendent, Naval Postgraduate School, Monterey	
	(Library)	(Copy 27)
54	Chief of Naval Research	
	(ONR 211 - Mr. Siegel)	(Copy 28)
	(ONR 438 - Mr. Cooper)	(Copy 29)
	(ONR 211 - Mr. Whitehead)	(Copy 30)
---	Stevens Institute of Technology, Davidson Laboratory, Hoboken, NJ 07030	
	(Library)	(Copy 31)
	(Mr. Al Strumpf)	(Copy 32)
11	Applied Research Laboratory, Penn State University	
	(Library)	(Copy 33)
	(Mr. W. R. Hall)	(Copy 34)
19	Applied Research Laboratory, University of Texas	
	(Library)	(Copy 35)
---	Applied Physics Laboratory, Johns Hopkins University, Charles & 34th St., Baltimore, MD 21218	
	(Library)	(Copy 36)
	(W. Venezia)	(Copy 37)
---	Virginia Polytechnical Institute, Blacksburg, VA 24060	
	(Library)	(Copy 38)
---	Massachusetts Institute of Technology, Cambridge, MA 02139	
	(Library)	(Copy 39)
	(Mr. Abkowitz)	(Copy 40)
---	Library of Congress, Washington, DC	
	(Science & Technology Library)	(Copy 41)
340	Director, Woods Hole Oceanographic Institute	(Copy 42)
302	Director, Scripps Institute of Oceanography	(Copy 43)
---	Commanding Officer, Naval Oceans Research & Development Activity, Bay St. Louis, MS 39520	(Copy 44)
---	Society of Naval Architects and Marine Engineers (SNAME), 74 Trinity Pl., New York, NY 10006	(Copy 45)
77	Director of Defense Research & Engineering, Washington	(Copy 46)
6	Director, Advanced Research Projects Agency, Washington	(Copy 47)
---	The Analytical Sciences Corp., Six Jacob Way, Reading, MA 01867	
	(Library)	(Copy 48)
---	Systems Control Institute, 1801 Page Mill Rd., Palo Alto, CA 94304	
	(Library)	(Copy 49)
75	Director, Defense Documentation Center	(Copies 50-61)

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER
NCSL TM-204-78

2. GOVT ACCESSION NO.

3. RECIPIENT'S CATALOG NUMBER

4. TITLE (and Subtitle)
6 Theory and Computer Program for Transfer Function Identification by the GRAM Identifier with Application to Undersea Vehicles.

5. TYPE OF REPORT & PERIOD COVERED
9 Technical Memo[redacted];

6. PERFORMING ORG. REPORT NUMBER

7. AUTHOR(s)
10 V. K./Jain,
G. J./Dobeck
L. J./Lawdermilt

8. CONTRACT OR GRANT NUMBER(s)
15 N61331-75-C-ØØ12
N61339-75-C-0122

9. PERFORMING ORGANIZATION NAME AND ADDRESS
University of South Florida
Tampa, Florida 33620

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
17 Task Area No. ZF 61112ØØ1,
Task Area No. SMW Ø2

11. CONTROLLING OFFICE NAME AND ADDRESS
Naval Coastal Systems Laboratory
Panama City, Florida 32407

12. REPORT DATE
11 [redacted]78

13. NUMBER OF PAGES
105 12 106 p.

14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)
18 NCSL 19 TM-2Ø4-78

15. SECURITY CLASS. (of this report)
Unclassified

15a. DECLASSIFICATION/DOWNGRADING SCHEDULE
N/A

16. DISTRIBUTION STATEMENT (of this Report)
Approved for Public Release - Distribution Unlimited
16 F61112, SMWØ2

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Computer Programs
Underwater Vehicles
Vehicles
Transfer Functions
Models

Dynamics
Identifiers:
Parameters Estimates
Measurement Filters
GRAM Identifier

Flight Test Data
Submerged Vehicles
Vehicle Dynamics

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

A method for the analysis of submerged vehicle dynamics is described. It enables the test engineer to identify the transfer function parameters from actual flight-test data (for post-flight analysis), or from simulated trajectories (for designing tow-test maneuvers), via the computer program GRAM. The method is noniterative and yields reliable estimates in the presence of disturbances and instrumentation noise.

361 520

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

I. INTRODUCTION

A major impediment in the study of the dynamics of a particular submerged vehicle is a lack of accurate functional relationship between the motion variables and the control inputs. In the uncoupled linear case such relationships may consist of linear transfer functions. Although their general form including the degrees of the denominator and numerator polynominals is known [1], the coefficients of these transfer functions are often unknown. They must be determined either through analytical formulas involving hydrodynamic coefficients or through identification algorithms performed upon experimental flight test data. The purpose of this report is to present a new identification method 'GRAM Identifier' and a computer program for its application to flight test data of submerged vehicles.

The method discussed possesses the following advantages: a) it is noniterative and therefore computationally fast, and b) it is noise-worthy [2]-[4]. Only the single-input, single-output case is considered in the present report. It will therefore be assumed that the flight test data consist of single input maneuvers, each caused by the actuation of a single control surface while the remaining control surfaces are held at zero deflection. To aid the engineer, a computer program entitled GRAM has been written that performs the necessary computations. The program is suitable for analysis of actual flight test data as well as for a simulation mode. In the latter case the flight trajectories are first generated, incorporating synthetic disturbance and measurement noise, and identification is then performed on the simulated trajectories. The simulation mode is useful when, for example, the approximate transfer functions of the vehicle are known (from hydrodynamic computations) and it is desired to find efficient maneuvers so as to develop a flight test plan.

The structure of the report is as follows. Section II presents the theory of the GRAM Identifier. Section III gives a user oriented description of the computer program. The results of some case studies, including those performed on actual vehicles, are provided in Section IV. Appendix A includes the listing and flow charts of the subroutines used by GRAM. Appendix B provides a brief discussion on the equivalence of z-domain and s-domain transfer functions and Appendix C deals with the solution of a key equation.

II. GRAM IDENTIFIER

The identification problem is formulated with reference to Fig. 1. The variable u represents a nonzero input variable -- the stern-plane angle, the rudder angle or other control surface deflection. The corresponding response y represents one of the motion variables -- pitch angle, yaw rate or some other. In part (a) of the figure is shown the vehicle, the instrumentation for the input-output variables, and the necessary samplers for digitization of these signals. Part (b) of the figure provides a discrete-time interpretation of the identification problem, which is stated as follows:

Given

i) the input and output measurements v(k), x(k), k=1,...,K,
ii) the integers n and r in the model

$$y(k) + a_1 y(k-1) + \ldots. + a_n y(k-n) = b_o u(k) + \ldots. + b_r u(k-r) \qquad (1)$$

iii) a statistical description of the noise processes w(k) and q(k), find the unknown parameters a_i and b_i so that the model provides the best fit (in some sense) into the measured data.

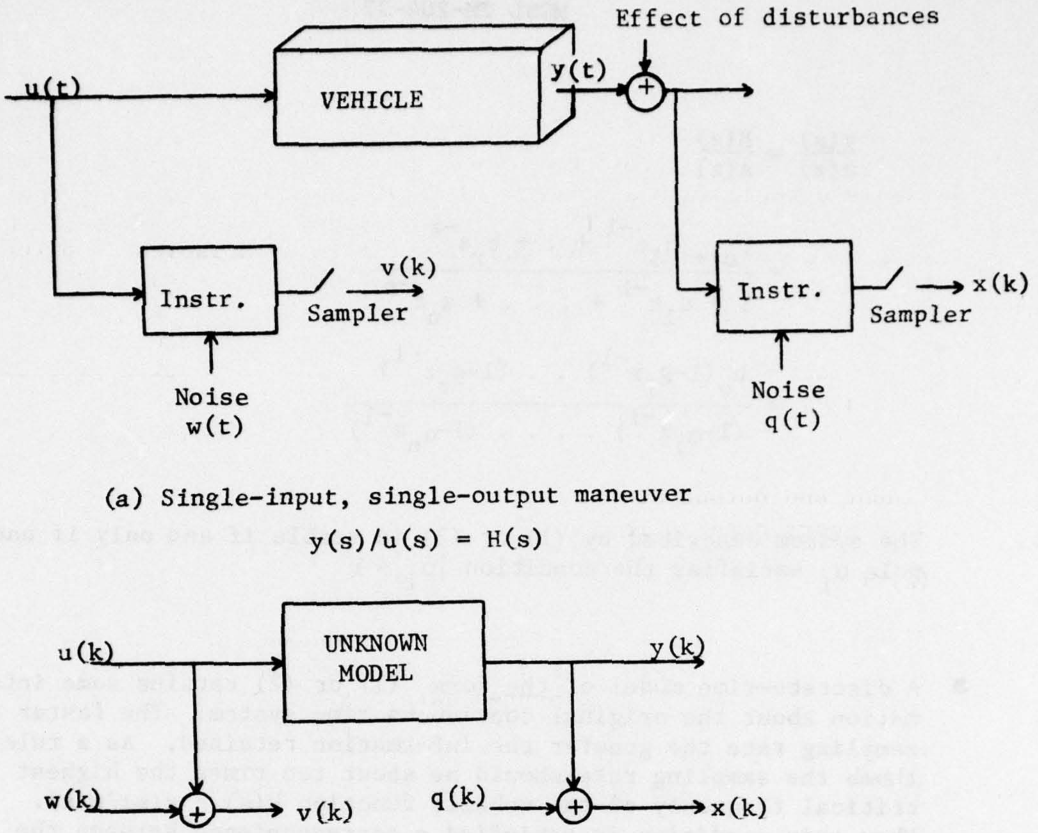
Note that the quantities a_i, b_i, y(k) and u(k) are in fact to be estimated. Only x(k) and v(k) are directly available.

Remarks

The reader familiar with the principles of signal sampling may wish to skip these remarks.

- Note that $y(k) = y(k^{\Delta})$, $u(k) = u(k\Delta)$, etc., where Δ is the sampling interval.

- In terms of the z-transform variable the relationship of (1) can be written as [5]



(a) Single-input, single-output maneuver

$y(s)/u(s) = H(s)$

(b) Discrete time identification problem

$y(z)/u(z) = H(z)$

Fig. 1. Single-input, single-output identification problem.

$$\frac{y(z)}{u(z)} = \frac{B(z)}{A(z)} \tag{2a}$$

$$= \frac{b_o + b_1 z^{-1} + . + b_r z^{-r}}{1 + a_1 z^{-1} + . . . + a_n z^{-n}} \tag{2b}$$

$$= \frac{b_o(1-\beta_1 z^{-1}) . . (1-\beta_r z^{-1})}{(1-\alpha_1 z^{-1}) . . . . (1-\alpha_n z^{-1})} \tag{2c}$$

The system described by (1, or (2) is stable if and only if each pole α_i satisfies the condition $|\alpha_i| < 1$

- A discrete-time model of the form (1) or (2) retains some information about the original continuous time system. The faster the sampling rate the greater the information retained. As a rule of thumb the sampling rate should be about ten times the highest critical frequency of the vehicle function $H(s) = y(s)/u(s)$. When this condition is satisfied a correspondence between the z-domain and s-domain functions may be achieved. Specifically, the equivalent continuous-time model becomes (see also Appendix B)

$$H(s) = \frac{\delta_o \, (s+q_1) . . (s+q_r)}{(s+p_1)(s+p_2) . . . . (s+p_n)} \tag{3}$$

where

$$q_i = -\frac{1}{\Delta} \ln (\beta_i) \qquad (\text{or } \beta_i = e^{-q_i \Delta})$$

$$p_i = -\frac{1}{\Delta} \ln (\alpha_i) \qquad (\text{or } \alpha_i = e^{-p_i \Delta})$$

The relationship in (1), or equivalently (2b), may be written as

$$A^T \xi_n \, y(z) = B^T \xi_r \, u(z) \tag{4}$$

where

$$A^T = [1 \; a_1 \; . . . . \; a_n]$$

$$B^T = [b_o \; b_1 \; . . . \; b_r]$$

$$\xi_n^T = [1 \; z^{-1} \; . \; . \; . \; . \; . \; . \; . \; . \; z^{-n}]$$

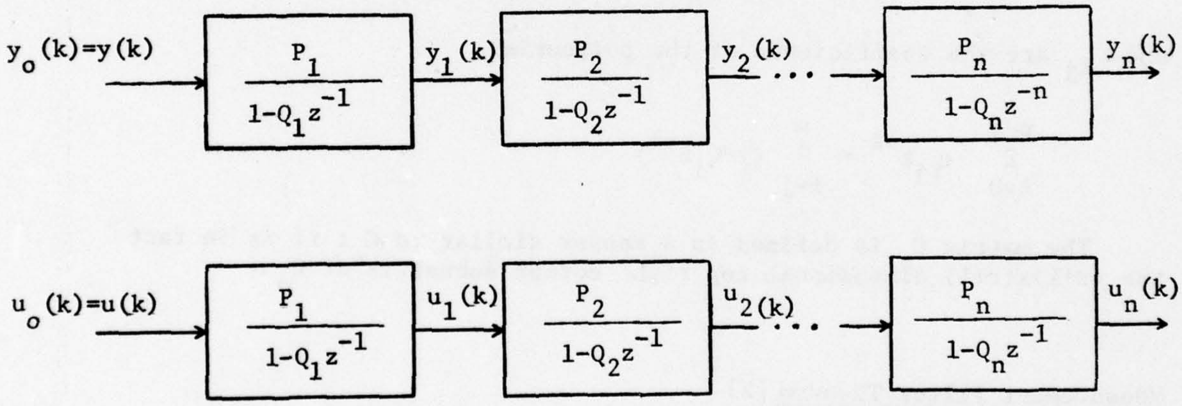
$$\xi_r^T = [1 \; z^{-1} \; . \; . \; . \; z^{-r}]$$

The super T denotes the transpose of a vector or matrix. Equation (4) should form the basis for modeling the vehicel dynamics. However, as discussed later, the use of the vector signals $\xi_n y(z)$, $\xi_r u(z)$ leads to poor results in identification. Instead GRAM relies upon certain measurement signals shown in Fig. 2 generated by means of first order digital filters. Specifically, use is made of the vector signals

$$Y(k) = [y_o(k), \; y_1(k), . \; . \; . \; . \; , \; y_n(k)]^T$$

$$U(k) = [u_{n-r}(k), \; . \; . \; . \; , \; u_n(k)]^T$$

consisting of the measurements at time instant kΔ. They will be called output and input measurement vectors, respectively. Their z-transforms are denoted as Y(z) and U(z). It can then be shown that


$y_o(k)=y(k)$
$\frac{P_1}{1-Q_1 z^{-1}}$
$y_1(k)$
$\frac{P_2}{1-Q_2 z^{-1}}$
$y_2(k)$ ...
$\frac{P_n}{1-Q_n z^{-n}}$
$y_n(k)$
$u_o(k)=u(k)$
$\frac{P_1}{1-Q_1 z^{-1}}$
$u_1(k)$
$\frac{P_2}{1-Q_2 z^{-1}}$
$u_2(k)$ ...
$\frac{P_n}{1-Q_n z^{-1}}$
$u_n(k)$

Output measurement sequences: $y_o(k), \; . \; . \; . \; . \; . \; , y_n(k)$

Input measurement sequences: $u_{n-r}(k), \; . \; . \; . \; , u_n(k)$

Fig. 2. Measurement filter system

$$Y(z) = \frac{1}{d_n(z)} \; C_n^T \xi_n y(z) \tag{5a}$$

$$U(z) = \frac{1}{d_n(z)} \; C_r^T \xi_r u(z) \tag{5b}$$

where

$$d_n(z) = \prod_{i=1}^{n} (1-Q_i z^{-i})/P_i$$

$$C_n = \begin{bmatrix} c_{00} & c_{01} & \cdots & c_{0n} \\ c_{10} & c_{11} & & 0 \\ \cdot & \cdot & & 0 \\ \cdot & \cdot & & \cdot \\ \cdot & \cdot & & \cdot \\ c_{n0} & 0 & & 0 \end{bmatrix}$$

and $c_{\ell j}$ are the coefficients of the polynomial

$$\sum_{\ell=0}^{n-j} c_{\ell j} z^{-\ell} = \prod_{i=j}^{n} (1-Q_i z^{-1})$$

The matrix C_r is defined in a manner similar to C_n; it is in fact the (r+1)x(r+1) dimensional top right corner submatrix of C_n.

Measurement Filter Theorem [2]

If the signals y(k) and u(k) satisfy (4) for some parameter vector A and B, then the measurement vectors satisfy the orthogonality condition

$$[\alpha^T - \beta^T] \begin{bmatrix} Y(k) \\ U(k) \end{bmatrix} = 0 \qquad \text{for all } k$$

where

$$\alpha = C_n^{-1} A \qquad (6a)$$

$$\beta = C_r^{-1} B \qquad (6b)$$

Proof The matrices C_n and C_r are upper triangular about the cross-diagonal, the latter having nonzero entries. Hence these matrices are nonsingular. We can therefore rewrite (4) as

$$(C_n^{-1}A)^T C_n^T \xi_n y(z) = (C_r^{-1}B) C_r^T \xi_r u(z)$$

Substituting 6a and 6b and dividing through by d(z) one has

$$\alpha^T \frac{1}{d(z)} C_n^T \xi_n y(z) - \beta^T \frac{1}{d(z)} C_r^T \xi_r u(z) = 0$$

Upon substituting (5a) and (5b) this equation yields

$$[\alpha^T \; -\beta^T] \begin{bmatrix} Y(z) \\ U(z) \end{bmatrix} = 0$$

The result sought by the theorem is obtained immediately upon taking the inverse transform. QED

Corollary: Let

$\lambda = [\alpha^T \; -\beta^T]^T$ Synthetic parameter vector

and

$f(k) = [Y^T(k) \; U^T(k)]^T$ Model-measurement vector

then

$$\lambda^T f(k) = 0 \qquad \text{for all } k \qquad (7)$$

Measurement vectors

As stated earlier, the sequences y(k) and u(k) are not actually available. Only x(k) and v(k) are, where

$$x(k) = y(k) + q(k)$$

$$v(k) = u(k) + w(k)$$

Because the system of measurement filters in Fig. 2. is linear, the following observation can now be made

Suppose that instead of processing y(k) the cascade of filters processes output noise q(k). Similarly, let the lower cascade of filters process the noise sequence w(k). And, let the resulting measurement sequences be denoted as $q_i(k)$ and $w_i(k)$, respectively.

Then

$$x_i(k) = y_i(k) + q_i(k)$$

$$v_i(k) = u_i(k) + w_i(k)$$

where $x_i(k)$ and $v_i(k)$ are the data-measurement sequences obtained by processing x(k) and v(k) by the two cascades of measurement filters.

Let

$$Q(k) = [q_o(k), ... , q_n(k)]$$

$$W(k) = [w_{n-r}(k), . . ., w_n(k)]$$

$$X(k) = [x_o(k), ... , x_n(k)]$$

$$V(k) = [v_{n-r}(k), . . ., v_n(k)]$$

$e(k) = [Q^T (k), W^T(k)]^T$ noise measurement vector

$g(k) = [X^T(k), V^T(k)]^T$ data measurement vector

Then

$$g(k) = f(k) + e(k)$$

For convenience, f(k), e(k) and g(k) will be called model-measurement vector, noise-measurement vector and data-measurement vector, respectively. To emphasize, the model-measurement vector f(k) is obtained by passing the model sequences y(k) and u(k) through the measurement filters; the noise-measurement vector e(k) by passing the noise sequences q(k) and v(k) through the same filters; and the data-measurement vector by passing the data sequences x(k) and v(k) through the system of mesurement filters.

Generalized Least-Squares Formulation of the Identification Problem

Given the data-measurement vectors g(k), k=1, . . ., K and the noise-measurement vector covariance

$$R = E \sum_{k=1}^{K} e(k)\, e^T(k) \qquad \text{(E: expected value operator)}$$

find the synthetic parameter vector λ that minimizes

$$J = \sum_{k=1}^{K} [g(k) - f(k)]^T R^{-1} [g(k)-f(k)] \tag{8}$$

under the constraint

$$\lambda^T f(k) = 0 \tag{9}$$

Remark

(The reader may wish to skip this in the first reading)

If q(k) and w(k) are stationary white noise processes with variances σ_q^2 and σ_w^2 respectively and cross-correlation coefficient ρ (which could of course be zero) then

$$R = \sum_{k=1}^{K} (K-k+1) \left[\begin{array}{c|c} \tilde{Q}(k)\tilde{Q}^T(k)\sigma_q^2 & \rho\tilde{Q}(k)\tilde{W}^T(k)\sigma_q\sigma_w \\ \hline \rho\tilde{W}(k)\tilde{Q}^T(k)\sigma_q\sigma_w & \tilde{W}(k)\tilde{W}^T(k)\sigma_w^2 \end{array}\right] \tag{10}$$

Here, $[\tilde{Q}(k),W(k)] = p(k)$ represents the measurement-vector sequences resulting from unit pulse ($\delta_k = \{1,0,0,\ldots\}$) stimuli at the measurement filter input terminals. We shall call p(k) the pulse-measurement vectors.

- Note that R has the form

$$R = \left[\begin{array}{c|c} R_{11}\sigma_q^2 & R_{12}\rho\sigma_q\sigma_w \\ \hline R_{21}\rho\sigma_q\sigma_w & R_{22}\sigma_w^2 \end{array}\right]$$

where the matrices R_{11}, $R_{12} = R_{21}^T$, and R_{22} are known (without the knowledge of σ_q^2, σ_w^2,andρ). They are determined entirely by the known measurement filters. When either σ_q or σ_w is zero, the matrix R becomes known up to a scalar multiple.

Solution of the Identification Problem

The solution λ and f(k) which minimize (8) under the constraint are obtained by the Lagrange multiplier method:

$$J^* = \sum_{k=1}^{K} ||g(k)-f(k)||^2_{R^{-1}} + \sum_{k=1}^{K} \nu_k(\lambda^T f(k)) \tag{11}$$

$$\frac{\partial J^*}{\partial f(k)} = -2R^{-1}(g(k)-f(k)) + \nu_k \lambda = 0 \tag{12a}$$

$$(g(k)-f(k)) = \frac{1}{2}\nu_k R \lambda \tag{12b}$$

$$\lambda^T (g(k)-f(k)) = \frac{1}{2}\nu_k \lambda^T R \lambda \tag{12c}$$

$$\frac{\partial J^*}{\partial \nu_k} = \lambda^T f(k) = 0 \tag{12d}$$

Equations (12c) and (12d) together yield

$$\lambda^T g(k) = \frac{\nu_k}{2} \lambda^T R \lambda$$
$$\frac{\nu_k}{2} = \frac{\lambda^T G(k)}{\lambda^T R \lambda} \tag{12e}$$

Substitution of (12b), (12d) and (12e) gives the minima with respect to f(k) and ν_k:

$$J^* = \sum_{k=1}^{K} \frac{\nu_k}{2} (g(k)-f(k))^T \lambda$$
$$= \sum_{k=1}^{K} \frac{\nu_k}{2} g^T(k)\lambda$$
$$= \sum_{k=1}^{K} \frac{\lambda^T g(k) g^T(k) \lambda}{\lambda^T R \lambda}$$

By defining the Gram matrix of the data-measurement vectors g(k)

$$G = \sum_{k=1}^{K} g(k) g^T(k) \tag{13}$$

we may write

$$J^* = \frac{\lambda^T G \lambda}{\lambda^T R \lambda} \tag{14}$$

The problem now is to minimize (14) with respect to λ. This is quite readily shown to be the eigenvector solution of

$$(G-\mu R)\lambda = 0 \tag{15}$$

corresponding to the smallest eignevalue μ_1.

Furthermore, it turns out that $J_{minimum} = \mu_1$ and f(k) are given by

$$f(k) = g(k) - \frac{\lambda^T g(k)}{\lambda^T R\lambda} R \lambda$$

Remarks

- The actual solution of the eigenvector problem in (15) is contingent upon the form of the R matrix. Three different cases arise which are discussed in Appendix C.

- The standard deviations σ_q and σ_w of the output and input noise sequences are frequently unavailable. Under white noise assumption they can be estimated approximately as follows. Suppose that the useful signal frequencies are limited to the frequency band $[0,f_1]$, then by high-pass filtering one can estimate the power-density of the noise in the region f_1 to $f_1 + f_2$; call this density $S(f_1)$. Suppose the standard deviation of the high-pass filtered signal is $\tilde{\sigma}$, then the standard deviation of the original noise signal may be estimated as

$$\hat{\sigma}^2 = 2f_1 S(f_1) + \tilde{\sigma}^2 \quad \text{and} \quad S(f_1) = \tilde{\sigma}^2/(2f_z)$$

- As mentioned earlier, a judicious choice of measurement filters can lead to rapid and successful identification of the vehicle transfer function. The primary consideration in this choice is that the resulting data-measurement sequences $y_i(k)$ should be as linearly independent as possible (so that the cross-correlations between them are as small as possible). For, if the measurement sequences were highly correlated, the matrix G would becomes ill-conditioned. Correspondingly, the solution $\hat{\lambda}$ would be unreliable. For example if the measurement filters were chosen to have $Q_i \simeq 0$ so as to have nearly all-pass characteristics (comparedto the vehicle's critical frequencies) than the resulting measurement sequences have pair-wise correlations approximating unity.

Another interesting case worth mentioning is that when each measurement filter, instead of being chosen as a recursive first order digital filter, is replaced by a unit delay. In this event the formulation coincides with that considered by Levin [6]. If the sampling rate for discretizing the motion variables is adequately high, which is usually true, the corresponding sequences $y_i(k) \equiv y(k-i)$ are highly correlated, rendering this choice undesirable [7]. Poor identification results therefore accrue.

- Since the measurement filters $(1-Q_i)/1-Q_i z^{-1})$ have low-pass frequency characteristics with unit d.c. gain, a convenient way to control the correlation between the resulting signals is to choose Q_i so that the mean power of measurement sequences diminish in a sensible manner. Call the mean power of the output of the ith filter as $\bar{p}_i$; the Q_i could be selected so that

$$\bar{p}_i \simeq \frac{n-i+1}{n+1} \bar{p}_o \qquad i = 1,2,\ldots, n$$

This choice of measurement filters has been implemented in the computer program GRAM as is available to the engineer on a select option basis

- As discussed earlier the minimum value achieved by the criterion function J is given by $\hat{\mu}$, the smallest eigenvalue of the equation posed in (15). This value will be called the <u>algorithm error</u>. However, since the engineer is interested really in the fidelity of output reconstruction a simple measure of fidelity may be used. Let $\hat{y}(k)$ be the reconstructed signal obtained by processing the measured input v(k) by the estimated transfer function (the true input u(k) is used when simulation mode is used in the computer program GRAM). Then the percent <u>reconstruction error</u> is defined as

$$\mathrm{ESR} = 100\sqrt{\sum_{k=1}^{K} (y(k) - \hat{y}(k))^2 / \sum_{k=1}^{K} y^2(k)}$$

III. PROGRAM DESCRIPTION

The purpose of this FORTRAN program is to determine a linear model from flight-test data of a submerged vehicle. The method used is the 'GRAM Identifier' discussed in Section II. The computer program is designed to work under three different modes. When ISIM = 0 analysis of actual flight test data is performed; with ISIM = 1 and ISIM = 2 flight trajectories are first simulated and then identification is performed.

When ISIM is either one or two the input data is generated in subroutine FILLV where the type of control input is specified by the parameter *INPT*. The flight trajectory is simulated using the z-domain vehicle transfer function when ISIM = 1 (subroutine RESPON) or using the vehicle impulse response when ISIM = 2 (subroutine CONVOL). Once the flight trajectories are generated the program uses the facility of adding synthetic white Gaussian noise (subroutine CORUPT) to the simulated flight trajectories. Next the model identification is performed based upon the actual or simulated flight trajectory through the 'GRAM Identifier' method. The identified model is used to reconstruct a flight trajectory which is compared to the actual or simulated flight trajectory for analysis.

When ISIM = 1 it is possible also to simulate a feedback system as shown in Fig. 3 wherein the vehicle transfer function, the input function (via option parameter INPT), the compensator constants and the gain constant are specified to the program.

The last of these is not read via data cards; it is entered directly in the subroutine RESPON. Identification is then performed upon the vehicle input-output. For reconstruction one may use open-loop reconstruction or closed-loop reconstruction.

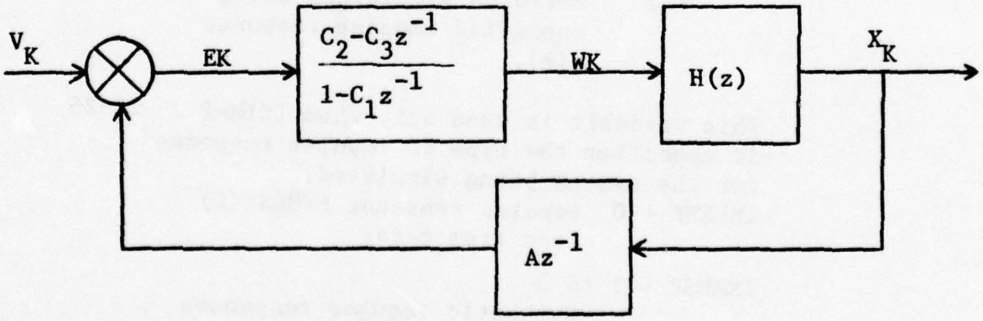


Fig. 3. Simulated feedback loop; $C_i \equiv$ COMPS(I), A = GAIN

The input data cards on the subsequent pages give a description of all input variables, and in so doing provide an understanding of the program use.

INPUT DATA CARDS

<u>CARD # 1</u> The first card is a title card. Columns 1 through 80 are available for an alphanumeric title.

<u>CARD # 2</u> First option card which contains eight variables.

Variable Name (Format)	Description	Columns	Preferred Value (if any)
N (I5)	Order of system	1-5	-
MP1 (I5)	Number of data points	6-10	-

IPLT
(I5)
	Plotter option;
	IPLT = 0 No plots
	= 1 Plots only on line printer
	= 2 Plots on printer and CALCOMP plotter
		11-15	1

ISIM (I5)	Simulation option ISIM = 0 Performs identification upon flight test data = 1 Performs simulation using z-domain transfer function coefficients = 2 Performs simulation using specified impulse response h(k).	16-20	-
IMRESP (I5)	This variable is used only when ISIM=2 it specifies the type of impulse response for the system being simulated. IMRESP = 0 Impulse response HPULSE(k) read from cards IMRESP = 1 to 5 Synthetic impulse responses generated (see subroutine CONVOL)	21-25	-
NPULSE (I5)	Number of impulse response points	26-30	-

INORM	This option allows various matrices to be normalized if INORM = 1	31-35	1

CARD # 3 through card # IX

$$IX = \begin{cases} 2 + 2*[MP1/8] & \text{if } ISIM = 0 \\ 4 + 2*N & \text{if } ISIM= 1 \\ 2 + [NPULSE/8] & \text{if } ISIM= 2 \end{cases}$$

[X] is the function that rounds off to the nearest integer greater than or equal to X.

These cards contain different types of data which is determined according to the option ISIM. Specifically if ISIM = 0 the output input data is placed in these cards. This information is placed on the cards in 8F10.1 fields with the odd positions containing the output data and even positions containing the input data. If ISIM = 1 these cards contain the z-domain transfer function coefficients. The number of coefficients must equal 2N+2 and must be entered as follows. The z-domain transfer function is of the form shown below

$$H(z) = \frac{b_1+b_2z^{-1}+b_3z^{-2}+....+b_{n+1}z^{-n}}{1+a_2z^{-1}+a_3z^{-2}+....+a_{n+1}z^{-n}} = \frac{NUM(z)}{DENOM(z)}$$

The coefficients should be entered each on a separate card in a D22.15 format in the following order.

$$1,\ a_2,\ a_3,....,a_{n+1},-b_1,\ -b_2,\ -b_3,....,-b_{n+1}$$

If ISIM=2 and IMRESP=0 these cards contain the impulse response. The length of impulse response is determined by the option NPULSE. Eight data points are placed on each card in an 8F10.0 format.

CARD # IX + 1 Second option card contains sixteen variables

Variable Name (Format)	Description	Columns	Preferred Value
INPT	Option to select a specific input sequence (used only when ISIM=1 or 2)	1-2	-

INPT = 1 Impulse
= 2 Step
= 3 Doublet
= 4 Squarewave

	INPUT = 5 Square wave followed by exponential = 6 Exponential = 7 Periodic impulse = 8 Triangular wave = 9 Exponential + decaying sinusoid = 10 Random noise = 11-20 shifted functions 1-10		
IREM (I2)	Option used to describe the order of the numerator compared to the denominator. This parameter controls the numerator *of the z-domain* model transfer function. Specifically, it limits the numerator degree in z^{-1} to N-IREM. For example if IREM = 1, then the model seeks a numerator. $NUM(z) = b_1 + b_2z^{-1}+....+b_nz^{-(n-1)}$	3-4	1
IZTS (I2)	This option determines the type of z domain to s domain transformation that is performed. IZTS = 0 Z domain to S domain conversion is not performed. That is an equivalent continuous time system is not found = 1 An equivalent continuous time system is found (from the discrete time transfer function $\hat{H}(z)$ based on a logarithmic z to s transformation). = 2 An equivalent continuous time system is found (from the discrete time transfer function H(z) based on a pulse delayed z to s transformation.	5-6	-
QOPT (I2)	Option used to determine measurement filter pole(s). If QOPT = 0 each of the measurement filter poles is set equal to the data value read as QSAV. If QOPT=1 the measurement filter poles are calculated in subroutine FINDQ.	7-8	-

FDBACK (I2)	This option allows a negative feedback path to be added to simulate a feedback system for which the vehicle is the plant. FDBACK = 0 No feedback = 1 Feedback is simulated	9-10	-
FDREC	FDREC is the variable to determine the type of reconstruction desired. FDREC = 0 Open loop reconstruction = 1 Closed loop reconstruction (Note FDREC must equal zero if FDBACK equals zero)	11-12	-
ILEVIN	This option is used when the LEVIN identification technique is desired. ILEVIN = 0 Gram identification technique performed. = 1 Levin identification technique performed.	13-14	0
IDLY	Delay introduced on input numerator $NUM(z) = z^{-IDLY}(b_1+....+b_{(n+1-IREM)}{}^{*}z^{-(N-IREM)})$	15-16	0
N (I5)	Order of model	21-25	-
MP1 (I5)	Number of data points	26-30	-
NPRD (I5)	Time scale parameter for input signal (useful only when ISIM = 1 or 2)	31-35	-
ISKIP (I5)	This variable determines the sequence of points plotted on the printer. If ISKIP = 1 every data point is plotted and if ISKIP = 5 every fifth point is plotted,etc.	36-40	-
DELTA (F5.0)	Sampling interval	41-45	-

QSAV (F5.0)	QSAV is the measurement filter pole(s) used only if QOPT = 0 and disregarded if QOPT = 1.	46-50	0.8 to 0.95
NSPQ (F5.0)	Noise to signal power ratio of the output sequence (used only if ISIM = 1 or 2).	51-55	0.01
NSPW (F5.0)	Noise to signal power ratio of the input sequence (used only if ISIM = 1 or 2)	56-60	0.01

CARD # IX + 2

This card contains the coefficients of a first forder compensator in the forward path preceding the vehicle. The general form of the compensator is;

$$C(z) = \frac{COMPS(2) - COMPS(3)z^{-1}}{1 - COMPS(1)z^{-1}} \quad \left(\simeq \frac{cs+b}{s+a}\right)$$

The coefficients are read in the form of 3F10.1 fields. If no compensation is desired a blank card should be inserted in this position.

END OF FILE CARD

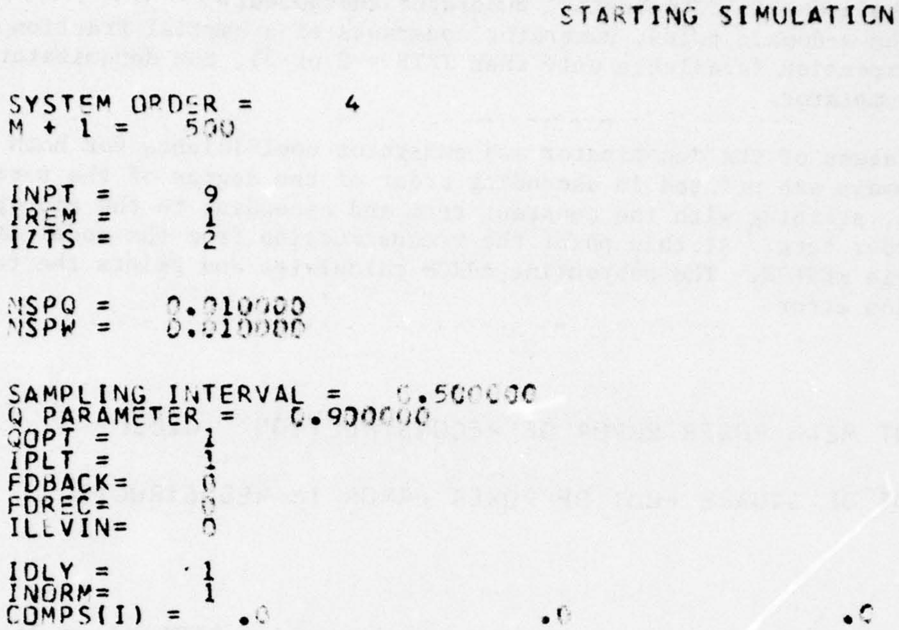
// Card on IBM 360 system.

MEMORY

The total storage required for the program is 152K bytes, or approximately 40K words, on an IBM 360/75 computer system. This will, of course, change if the array dimensions are changed to meet the users test requirements. Presently the program can accept up to one thousand data points each for the input and output signals. The model sought (or the simulated model entered when ISIM = 1) can be as high as ninth order. Under the third simulation mode (ISIM = 2) the impulse response can be of a length up to sixty four data points. It is important to note that the square matrices G and Z and the vectors GAMMA, XLAMDA, and COEFF should have a dimension at least as large as (N+N+2) where N is the order of the model.

OUTPUT

The first line of output is the title which is followed by a column of program variables as follows:



```
                                         STARTING SIMULATICN


SYSTEM ORDER =      4
M + 1 =    500

INPT =      9
IREM =      1
IZTS =      2

NSPQ =    0.010000
NSPW =    0.010000

SAMPLING INTERVAL =    0.500000
Q PARAMETER =    0.900000
QOPT =      1
IPLT =      1
FDBACK=     0
FDREC=      0
ILLVIN=     0

IDLY =      1
INORM=      1
COMPS(I) =    .0                     .0                     .0
```

The next portion of the output is a plot of the vehicle input and output. On the left hand side three columns of printout list the serial number of the data point, the instantaneous value of the output and the instantaneous value of the input repectively. In case a feedback loop is employed (i.e. FDBACK = 1) then the command input (i.e. the input to the feedback system) is plotted preceding the vehicle input-output plot. Next the following title is printed

GRAM IDENTIFIER

The next output line lists the values of measurement filter poles (Q(I)). Note that N measurement filters are used where N is the system order. All Q(I) are equal if QOPT = 0.

Following the Q parameters is the listing of the gram matrix G. This is an NPNP2 x NPNP2 matrix where NPNP2 = N+N+2. The item printed next is the noise correction matrix Z which is generated in the subroutine BUILDZ. This matrix is also NPNP2 x NPNP2 dimensional. The next line of printout is the Synthetic Coefficient Vector XLAMDA which is generated using the subroutines SOLVE1, 2, and 3. Following the Synthetic Coefficient Vector is the transformation matrix A (again an NPNP2 x NPNP2 matrix) which is premultiplied with XLAMDA to obtain the desired parameter vector GAMMA. It is generated in the subroutine BUILDA. The next value printed is the estimate of bias in data followed by the variable NN. At this point in the output the following are printed

a) the z-domain denominator, numerator and poles,

b) the s-domain poles, numerator constants of a partial fraction expansion (available only when IZTS = 2 or 3), the denominator and numerator.

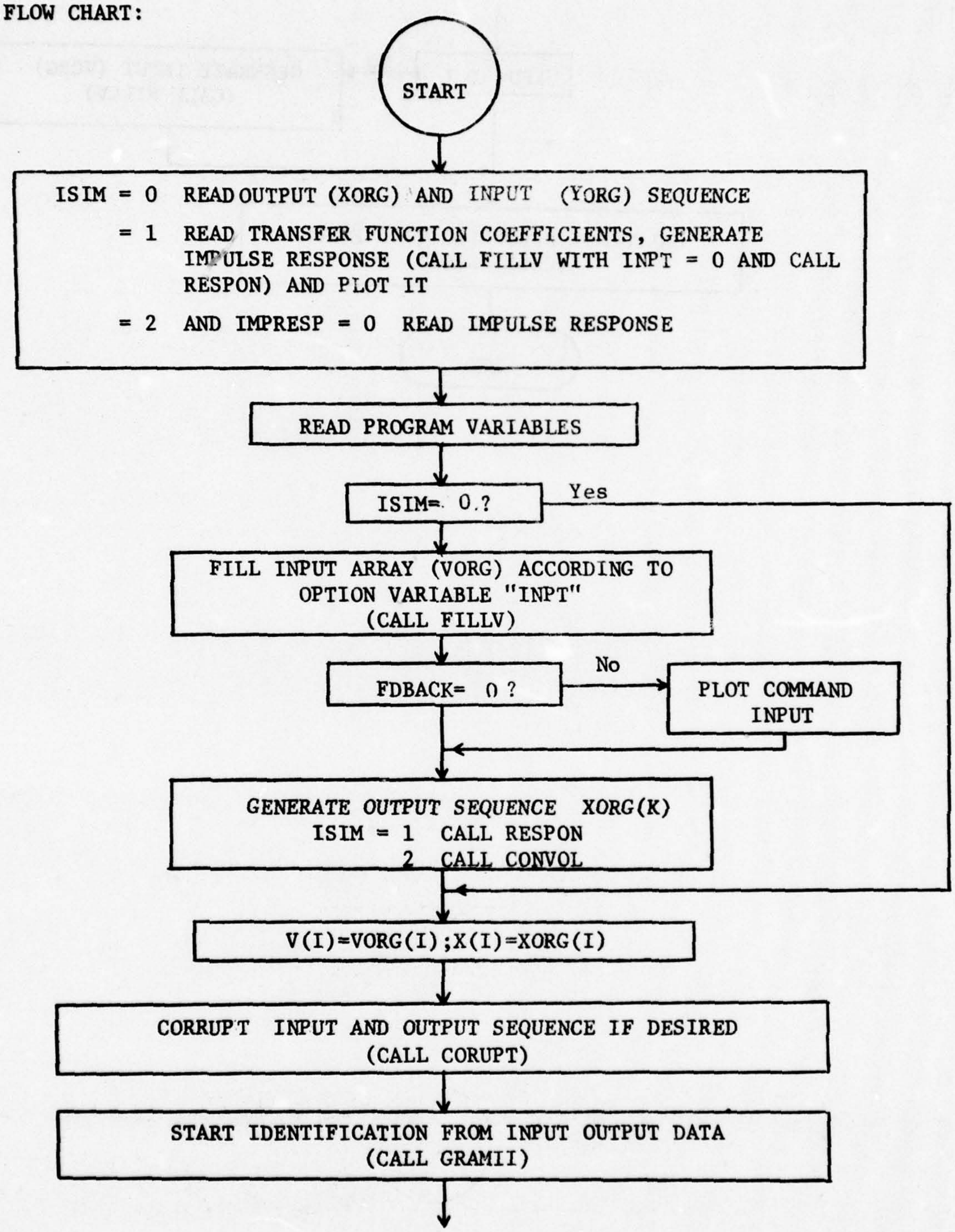
The values of the denominator and numerator coefficients for both the Z and S-domain are printed in ascending order of the degree of the term it multiplies, starting with the constant term and ascending to the appropriate highest order term. At this point the reconstruction from the model is obtained via RESPON. The subroutine ERROR calculates and prints the reconstruction error

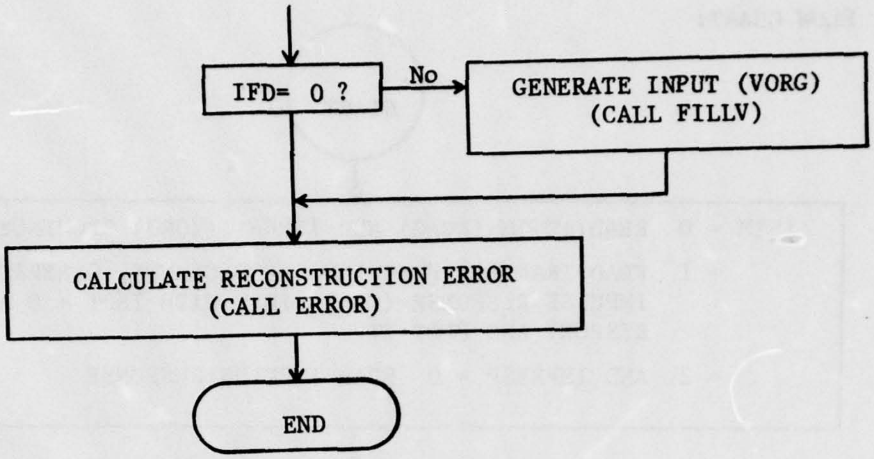
```
PER CENT MEAN POWER ERROR OF RECONSTRUCTION    0.000

PER CENT OF SQUARE ROOT OF POWER ERROR IN RECOSTRUCTION    0.002
```

The last output is the plot of the true response (when ISIM = 1 or 2) or actual flight test data when ISIM = 0 and the reconstructed response. The same format is used as the previous one for the vehicle input output plot.

PROGRAM GRAM

FLOW CHART:


START
ISIM = 0 READ OUTPUT (XORG) AND INPUT (YORG) SEQUENCE
= 1 READ TRANSFER FUNCTION COEFFICIENTS, GENERATE IMPULSE RESPONSE (CALL FILLV WITH INPT = 0 AND CALL RESPON) AND PLOT IT
= 2 AND IMPRESP = 0 READ IMPULSE RESPONSE
READ PROGRAM VARIABLES
ISIM= 0.?
Yes
FILL INPUT ARRAY (VORG) ACCORDING TO OPTION VARIABLE "INPT" (CALL FILLV)
FDBACK= 0 ?
No
PLOT COMMAND INPUT
GENERATE OUTPUT SEQUENCE XORG(K)
ISIM = 1 CALL RESPON
2 CALL CONVOL
V(I)=VORG(I);X(I)=XORG(I)
CORRUPT INPUT AND OUTPUT SEQUENCE IF DESIRED (CALL CORUPT)
START IDENTIFICATION FROM INPUT OUTPUT DATA (CALL GRAMII)


IFD= 0 ?
No
GENERATE INPUT (VORG)
(CALL FILLV)
CALCULATE RECONSTRUCTION ERROR
(CALL ERROR)
END

```
C       PROGRAM GRAM
        DIMENSION X(1000),V(1000),XORG(1C00),VORG(1000),XREC(1000)
        DIMENSION DATA(1000,2),DATA2(1C00,2),BUFF(3000)
        DIMENSION G(20,20),Z(20,20),GAMMA(20),XLAMDA(20),COEFF(20)
        DIMENSION HPULSE(64)
        DIMENSION TITLE(80)
        COMMON NN
        REAL*8 G,Z, GAMMA,XLAMDA,COEFF,COMPS
        REAL*8 DELTA,Q,QSAV,DELSAV,AVGQ,AVGW,SUMV2,XSAV
        REAL NSPQ,NSPW
        INTEGER QOPT,FDBACK,FDREC
        EQUIVALENCE (Z(1,1),BUFF(1)),(G(1,1),BUFF(1501))
        EQUIVALENCE (DATA(1,1),X(1)),(DATA(1,2),V(1))
        EQUIVALENCE (XORG(1),DATA2(1,1)),(XREC(1),DATA2(1,2))
        EQUIVALENCE (NSPQ,SIGQ),(NSPW,SIGW)
        COMMON /COMPEN/COMPS(1C)
        COMMON /GKRD/IGKR
C
        WRITE(6,1022)
        READ(5,1021)TITLE
        WRITE(6,1021)TITLE
        WRITE(6,1023)
        MAXPL=1000
        MAX=20
4320    READ(5,1001)    N,MP1,IPLT,ISIM,IMRESP,NPULSE,INORM
        NPNP2=N+N+2
        RDEL=0.01
        IF(ISIM.EQ.0)READ(5,6995)(XORG(K),K=1,MP1),(VORG(K),K=1,MP1)
        IF(ISIM.EQ.2.AND.IMRESP.EQ.0)READ(5,6995)(HPULSE(K),K=1,NPULSE)
        IF(ISIM.NE.1)GO TO 6622
C          READ DIFFERENCE EQUATION PARAMETERS
        READ(5,701,END=1234)(COEFF(I),I=1,NPNP2)
        CALL FILLV(VORG,MP1,0,NPRD)
        CALL RESPON(X,VORG,N,COEFF,XLAMDA,MP1,0)
        IF(IPLT .NE. 2) GO TO 6622
        CALL PLOP8(MP1,1,X,MAXPL,C.O,RDEL,3,
       117HIMPULSE RESPONSE_,
       216HTIME IN SECONDS_,BUFF)
6622    CONTINUE
        KKKK=0
C
C
4321    READ(5,1,END=1234)INPT,IREM,IZTS,QOPT,FDBACK,FDREC,ILEVIN,IDLY,
       1N DEN,MP1,NPRD,ISKIP,DELTA,QSAV,NSPQ,NSPW
        READ(5,6995)(COMPS(I),I=1,3)
        IF(N.EO.10000)GO TO 4320
        Q=QSAV
        WRITE(6,1000)N,MP1,INPT,IREM,IZTS,NSPQ,NSPW,DELTA,Q,COPT,
       1IPLT,FDBACK,FDREC,ILEVIN,IGKR,IDLY,INORM,   LCOMPS(I),I=1,3)
        NM1=N-1
        NP1=N+1
        NP2=N+2
        NPNP1=N+N+1
        NPNP2=N+N+2
        RHO=0.0
        NN=N-IREM
```

```
      IF(ISIM.EQ.0)GO TO 23
C
C        FILLING THE INPUT ARRAY ACCORDING TO OPTION PARAMETER, INPT
      CALL FILLV(VORG,MP1,INPT,NPRO)
      IF(FDBACK.EQ.0)GO TO 6626
      DO22I=1,MP1
      V(I)=VORG(I)
22    X(I)=0.0
      IF(KKKK.NE.0) GO TO 6616
      CALL PLOTIT(DATA ,2,MP1,1,MP1,ISKIP,MAXPL,1,1.0)
6616  CONTINUE
      IF(IPLT.NE.2  .OR.  KKKK.GT.0) GO TO 6626
      CALL PLOP8(MP1,2,DATA ,MAXPL,0.0,RDEL,3,
     125HINPUT TO FEEDBACK SYSTEM_,
     216HTIME IN SECONDS_,BUFF)
6626  CONTINUE
C
C        GENERATING SEQUENCE X(K)
      IF(ISIM.EQ.1)CALL RESPCN(XORG,VORG,N,COEFF,XLAMDA,MP1,FDBACK)
      IF(ISIM.EQ.2)CALL CONVCL(HPULSE,VORG,XORG,NPULSE,MP1,IMRESP)
23    DO24I=1,MP1
      V(I)=VORG(I)
24    X(I)=XORG(I)
      IF((NSPQ+NSPW)*ISIM .NE.0)CALL CORUPT(X,V,SIGQ,SIGW,MP1)
      IF(KKKK.NE.0) GO TO 6611
      WRITE(6,1003)
      CALL PLOTIT(DATA ,2,MP1,1,MP1,ISKIP,MAXPL,1,1.0)
6611  CONTINUE
      IF(IPLT.NE.2  .OR.  KKKK.GT.0) GC TO 6633
      CALL PLOP8(MP1,2,DATA ,MAXPL,0.0,RDEL,3,
     127HCORRUPTED INPUT AND OUTPUT_,
     216HTIME IN SECONDS_,BUFF)
6633  CONTINUE
      N=NDEN
C     START IDENTIFICATION FROM INPUT OUTPUT DATA
C
      CALL GRAMII(X,V,MP1,SIGQ,SIGW,RHC,N,DELTA,Q,QOPT,IREM,IZTS,GAMMA,
     1XLAMDA,G,Z,MAX,ILEVIN,IDLY,INORM)
      IFD=0
      IF(FDBACK.GE.1  .AND.  FDREC.EQ.1) IFD=FDBACK
      IF(IFD.NE.0) CALL FILLV(VORG,MP1,INPT,NPRD)
      CALL ERROR(XREC,VORG,GAMMA,MP1,N,XLAMDA,XORG,IFD,IDLY)
C
C     PLOT RECONSTRUCTION
      WRITE(6,66)
      WRITE(6,1004)
      IF(IPLT.EQ.0) GO TO 6544
      CALL PLOTIT(DATA2,2,MP1,1,MP1,ISKIP,MAXPL,1,1.0)
6544  CONTINUE
99    RDEL=DELTA
      IF(IPLT.NE. 2) GO TO 6644
      CALL PLOP8(MP1,2,DATA2,MAXPL,0.0,RDEL,3,
     140HTRUE RESPONSE VS RECONSTRUCTED RESPONSE_,
     216HTIME IN SECONDS_,BUFF)
6644  CONTINUE
C
      KKKK=KKKK+1
100   GO TO 4321
1234  CALL PICSIZ(0,0,0.0)
1022  FORMAT(////////////////////////////////////////////////////////)
1021  FORMAT(80A1)
```

```
1023  FORMAT(/,1X,'****************************************************',
     1 '***************************************')
1001  FORMAT(8I5)
701   FORMAT(D22.15)
1     FORMAT(8I2,4X,4I5,4F5.0)
1003  FORMAT(//,5X,'OUTPUT (*) AND INPUT (+)',/)
1004  FORMAT(//,5X,'OUTPUT (*) AND RECONSTRUCTION (+)',/)
66    FORMAT(//,1X,'TRUE RESPONSE VERSES RECONSTRUCTED RESPONSE',//)
6995  FORMAT(8F10.0)
1000  FORMAT(1H1,50X,'STARTING SIMULATION',/20X,'SYSTEM ORDER = ',I5,/,
     120X,'M + 1 = ',I5,///,20X,'INPT = ',I5,/,20X,
     2'IREM = ',I5,/,20X,'IZTS = ',I5,///,20X,'NSPQ = ',F10.6,/,20X,
     3'NSPW = ',F10.6,///,20X,'SAMPLING INTERVAL = ',F10.6,/20X,'Q PARAM
     4ETER = ',F10.6,/,20X,'COPT = ',I5,/,20X,'IPLT = ',I5,/,20X,
     5'FDBACK=',I5,/,20X,'FDREC= ',I5,/20X,'ILEVIN=',I5,/20X,'IGKR = ',
     6I5,/20X,'IDLY = ',I5,/20X,'INORM= ',I5,/20X,'COMPS(I) = ',3G17.10,
     7/)
      STOP
C
C
C
C
C
C          DEFINITION OF PARAMETERS USED IN THE SIMULATION OF A
C          LINEAR DYNAMIC SYSTEM
C
C          X IS THE CORRUPTED OUTPUT SEQUENCE
C          V IS THE CORRUPTED INPUT SEQUENCE
C          GAMMA IS THE COEFFICIENT VECTOR
C
C          MAX = ACTUAL DIMENSION SIZE OF 2-DIM ARRAYS IN THE DIMENSION
C          STATEMENT
C          N = ORDER OF SYSTEM
C          THE MAXIMUM VALUE OF N IS MAX/2-1
C          MP1 = M+1, THE TOTAL NUMBER OF SAMPLED POINTS IN EACH SEQUENCE
C
C          SIGQ = THE STANDARD DEVIATION OF THE OUTPUT NOISE SEQUENCE,Q(K)
C          SIGW = THE STANDARD DEVIATION OF THE INPUT NOISE SEQUENCE, W(K)
C          HOWEVER IN THE READ STATEMENT THE DESIRED NOISE TO SIGNAL POWER RATIO
C          *******                                ***************************
C          IS READ INTO SIGQ AND SIGW FROM WHICH THE TRUE STANDARD DEVIATIONS
C          ARE COMPUTED AND STORED BACK INTO SIGQ AND SIGW
C          RHO = EXPECTATION( W(K)*Q(K) )
C
C          DELTA IS THE SAMPLING INTERVAL
C          Q IS THE DENOMINATOR PARAMETER OF THE KNOWN FIRST ORDER DIGITAL
C          FILTERS FOR THE GRAM II TECHNIQUE
C          QSAV IS THEIR CUTOFF FREQUENCY
C
C
C     IGKR=0  USE IS MADE OF THE FIRST ROW OF ADJOINT
C          1  DIAGONAL (NEGATIVE ENTRIES SET TO ZERO)
C          2  ABSOLUTE VALUE OF DIAGONAL
C
C
C
C     ISIM=0  READ EXPERIMENTALLY MEASURED INPUT OUTPUT DATA
C          1  READ COEFFICIENTS OF H(Z), THEN SIMULATE INPUT-OUTPUT
C          2  READ IMPULSE RESPONSE HPULSE(K), K=1,...,NPULSE AND SIMULATE
C
C
C          IREM = DEGREE OF DENOMINATOR MINUS DEGREE OF NUMERATOR
```

```
C         IDLY = DELAY INTRODUCED ON INPUT NUMERATOR; ZETA=1/Z
C                ZETA**IDLY ( B(1)+.. +B(N+1-IREM)*ZETA**(N-IREM)  )
C
C       IZTS = 0 PRINTING OF DISCRETE TIME TRANSFER FUNCTION ONLY AND
C               THE POLES OF THE Z DOMAIN
C            = 1 IF LOGARITHMIC TRANSFORMATION IS TO BE CALCULATED
C            = 2 IF DELAYED PULSE INVARIANT TRANSFORMATION IS TO BE CALCULATED
C
C       INPT=1  IMPULSE,    2: STEP,   3:  DOUBLET (DURATION NPUL)
C       4:    SQWAVE (PERIOD NPUL),    5:   SQ-EXP,   6:   EXP,   7:   PRO IMPL
C       8:    TRI WAVE,    9:   EXP+OSC,   10:   RANDOM
C
C         QOPT = 0 IF Q(I)=QSAV
C                1 IF Q(I) GENERATED IN QFIND
C
C         FOR AN UNSTABLE SYSTEM FEEDBACK MAY BE PROVIDED BY SETTING
C                FEEDBACK=1.  A COMPENSATOR IN THE FORWARD PATH IS
C                PROVIDED ON A   NONOPTIONAL   BASIS, EXCEPT THAT
C                WHEN THE COMPENSATOR CARD HAS ZERO ENTRIES THE PROGRAM
C                AUTOMATICALLY SETS THE COMPENSATOR TO C(Z)=1.0
C         FEEDBACK GAIN IS MANUALLY ENTERED IN SUBROUTINE RESPON
C         AS THE VARIABLE "GAIN"
C                ********THE COMPS(I) COEFFICIENTS
C                OF C(Z)= (COMPS(2)-COMPS(3)/Z) / (1 -COMPS(1)/Z)  MUST
C                MUST BE READ,  A BLANK CARD MAY BE PROVIDED IF
C                IF NO COMPENSATOR IS DESIRED
C        FDREC =0 IF OPEN LOOP RECONSTRUCTION IS DESIRED
C                MUST BE ZERO IF  FDBACK IS ZERO
C              1 IF CLOSED LOOP RECONSTRUCTION DESIRED
C
C
C         POLES OF THE CONTINUOUS DOMAIN MUST BE DISTINCT AND NON-ZERO
C         FOR TRANSFORMATION TO BE VALID
C
C         IT IS IMPORTANT TO NOTE THAT THE VALUES OF THE CONTINUOUS
C         SIGNALS SAMPLED AT TIME=(K-1)*DELTA ARE STORED IN THE KTH
C         SEQUENCE POSITION OF THE ARRAYS
C
C         DATA DECK CONSISTS OF A READ OPTION CARD (N,IPLT),
C         THE Z-DOMAIN COEFFICIENTS OF THE ORIGINAL TRANSFER FUNCTION,
C         AND A PARAMETER CARD(N,MP1,INPT,IGRM,IREM,ISTZ,SIGQ,SIGW,
C         DELTA,QSAV,QOPT,NPRD,FDBACK,FDREC).
C
C         PROGRAM READS SEVERAL PARAMETER CARDS.  LAST PARAMETER CARD
C         MUST HAVE 1 IN COL 1 TO READ ANOTHER TRANSFER FUNCTION AND
C         PARAMETER CARD SET.
C
C         IPLT=0   NO  PLOTS
C         IPLT=1   PLOTS ONLY WITH PRINTER
C         IPLT=2   PLOTS ON CALCOMP AS WELL AS PRINTER
C
C
C
C
C
      END
```

IV. APPLICATION EXAMPLES

Example 1

For a six-man submersible the transfer functions describing its dynamics were obtained from the hydrodynamic coefficients via a computer program RGEORGE. The pitch vs. stern-plane dynamics will be used here to demonstrate the application of GRAM Identifier (ILEVIN = 0 in the program). Specifically the transfer function relating these variables is

$$H_4(s) = \frac{-0.34320s^2 -0.17384s -0.008631}{s^4 + 1.47989s^3 + 0.11833s^2 + 0.02048s + 0.00102}$$

$$= \frac{-0.08348 + j0.48366}{s + 0.00919 + j0.11378} + \frac{-0.08348-j0.48366}{s + 0.00919-j0.11378}$$

$$+ \frac{0.16696}{s + 1.4057} + \frac{0.00002}{s + 0.05579}$$

For all practical purposes this is seen to be equivalent to

$$H_4(s)= H_3(s) = \frac{-0.34320s -0.15469}{s^3 + 1.58950s^2 + 0.26054s + 0.00306}$$

$$= \frac{-0.08349 + j0.48367}{s+0.00919+j0.11378} + \frac{-0.08349-j0.48367}{s+0.00919-j0.11378} + \frac{0.16696}{s+1.4057}$$

In this third order function the energy of the complex pole pair is 26.9 while the energy associated with the real pole is 0.01; the latter thus represents only 0.037% of the energy at the dominant complex pole pair. Therefore, unless the input is such that its spectral content is rich in radian frequencies around 1.4, this mode will be extremely feeble. We will in fact call this mode a micromode [3]. When this micromode is not appropriately excited the vehicle transfer function may be approximated as

$$H_4(s) \cong H_2(s) = \frac{-0.16698s + 0.10853}{s^2 + 0.1838s + 0.01303}$$

$$= \frac{-0.08349+j0.48367}{s+0.00919+j0.11378} + \frac{-0.08349-j0.48367}{s+0.00919-j0.11378}$$

Before describing the various experiments conducted, the z-domain description of the pitch transfer function is first provided. The transfer function $H_4(s)$ was transformed by the Leading-Edge-Pulse Equivalence method (Appendix B). A sampling interval $\Delta = 0.5$ second was used yielding

$$H_4(z) = \frac{(10^{-4})(0.20744z^{-1}-0.19065z^{-2}-0.15179z^{-3}+0.13714z^{-4})}{1-3.45527z^{-1}+4.38952z^{-2}-2.41135z^{-3}+0.47714z^{-4}}$$

In all of the five experiments performed 250 seconds of simulated data (MP1 = 500) were used. This represents approximately 2 1/2 time constants of the dominant mode.

Experiment 1

The function $H_4(z)$ is employed to simulate the discrete-time trajectory $\theta(k)$ to a given stern-plane input $\delta_s(k)$ (INPT = 9, NPUL = 100). As stated earlier $\theta(s)/\delta_s(s)$ is effectively a third order function, however, a fourth order identification was first performed without masking the data with noise (NSPW = NSPQ = 0.0). The identification yielded the following $\hat{H}_4(s)$ with the option variables chosen as ISIM = 1, IREM = 1, IDLY = 1, QOPT = 1 and IZTS = 2:

$$\hat{H}_4(s) = \frac{-0.34323s^2-0.17394s-0.00867}{s^4+1.48029s^2+0.11867s^2+0.02049s+0.00103}$$

$$\text{ESR} = 0.000 \qquad \text{(see page 12)}$$

Clearly this identification is good since the model found is almost identical to the given $H_4(s)$. A comparison of the poles of the given $H_4(s)$ and the identified poles is shown below.

$H_4(s)$ Poles	Identified Poles
$-0.00919 + j0.11378$	$-0.00919 + j0.11378$
$-0.00919 - j0.11378$	$-0.00919 - j0.11378$
-1.4057	-1.4058
-0.05579	-0.05603

Although the vehicle transfer function was identified perfectly, any quick conclusions as to the effectiveness of the identification method are misleading. Because even the slightest amount of noise on the data will mask the micro-micro-mode $(\frac{0.00002}{s+0.05579})$ making its identification impossible. Therefore the remaining experiments will pertain to third order identification except the last one. The latter is a second order run demonstrating the detection of the dominent pole pair.

Experiment 2

This run is identical to the previous one except that a third order model was sought (N = 3). The transfer function found was

$$\hat{H}_3(s) = \frac{-0.00435s^2-0.33603s-0.14986}{s^3+1.38026s^2+0.03806s+0.01774}$$

$$\text{ESR} = 0.447$$

The identified poles are compared to the true poles below

$H_3(s)$ Poles	Identified Poles
-0.00919 + j0.11378	-0.00919 + j0.11378
-0.00919 - j0.11378	-0.00919 - j0.11378
-1.4057	-1.36187

Experiment 3

Ten percent rms noise was added to both the input and output data (NSPW = NSPQ = $(0.1)^2$). A third order identification was performed using an input generated via option INPT = 9, NPUL = 100. The test failed due to the extremely poor spectral content of the input. Specifically the input signal did not have sufficient energy at radian frequencies around 1.4, consequently it did not properly excite the pole at that location. A different input was therefore used (INPT = 5, NPUL = 10, QOPT = 1). The input and output signals are shown in Fig. 4a. The corresponding results yielded by GRAM are as follows:

$$\hat{H}_3(s) = \frac{-0.50521s^2-2.75286s-2.5146}{s^3+1.76398s^2+0.04714s+0.02249}$$

$$ESR = 5.275$$

A comparison of the identified poles to the true poles is given below.

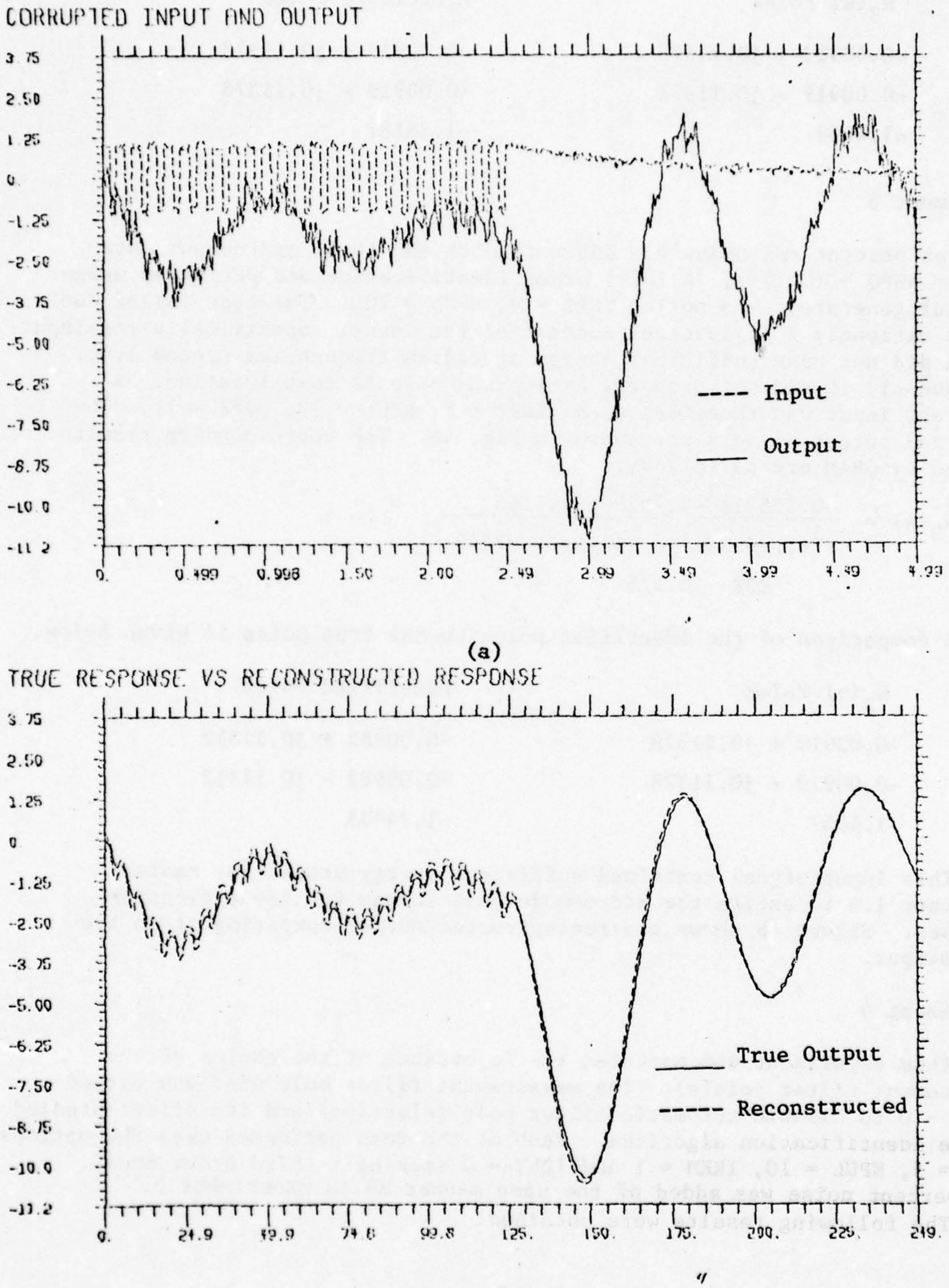
$H_3(s)$ Poles	Identified Poles
-0.00919 + j0.11378	-0.00982 + j0.11312
-0.00919 - j0.11378	-0.00982 - j0.11312
-1.4057	-1.74435

This input signal contained sufficient energy around the radian frequency 1.4 to excite the micro-mode just enough for identification purposes. Figure 4b shows the reconstructed output comparing it to the true output.

Experiment 4

This experiment demonstrates the importance of the choice of the measurement filter pole(s). The measurement filter pole QSAV was varied (QOPT = 0 to disable automatic filter pole selection) and its effect studied on the identification algorithm. Each of the runs performed uses the options INPT = 5, NPUL = 10, IREM = 1 and IDLY = 0 seeking a third order model. Ten percent noise was added of the same manner as in Experiment 3.

The following results were obtained:



(a)

(b)

Figure 4. Experiment 3 - Third Order pitch vs Stern-Plane Identification

QSAV	ESR (Percent Error to Signal ratio RMS)
0.70	∞
0.80	33.986
0.85	14.374
0.90	3.947
0.95	4.556
0.98	5.696

This example should make the user aware of the flexibility provided to the test engineer by the measurement filter pole(s). The value of the measurement filter pole should be such that each successive measurement filter attenuates the input signal by a reasonable fraction; in particular the output of the last measurement filter should not be an order of magnitude lower in power than the input signal to the first measurement filter. More information on the choice of measurement filters is available in reference [2].

Experiment 5

This final experiment demonstrates the detection of the dominant pole pair. The input (INPT = 6, NPUL = 200) and output signals were masked with 10% rms noise (NSPW = NSPQ = $(0.1)^2$) and the following option parameters were used: QSAV = 0.95 IREM = 1, IDLY = 0. The input used (INPT = 6) is an exponentially decaying function whose time constant, and therefore the cutoff of the power spectrum curve, is controlled by the value of NPUL (Time constant = NPUL * DELTA). Therefore to identify the slow poles (-0.00919 ± j0.11378) an input rich in radian frequencies around 0.114 is desired. A vlue of NPUL = 200 produces an adequate power spectrum. The input and output signals are shown in Fig. 5a. The computer program identified the second order mode as follows

$$\hat{H}_2(s) = \frac{0.02308s+0.09245}{s^2+0.01904s+0.01318}$$

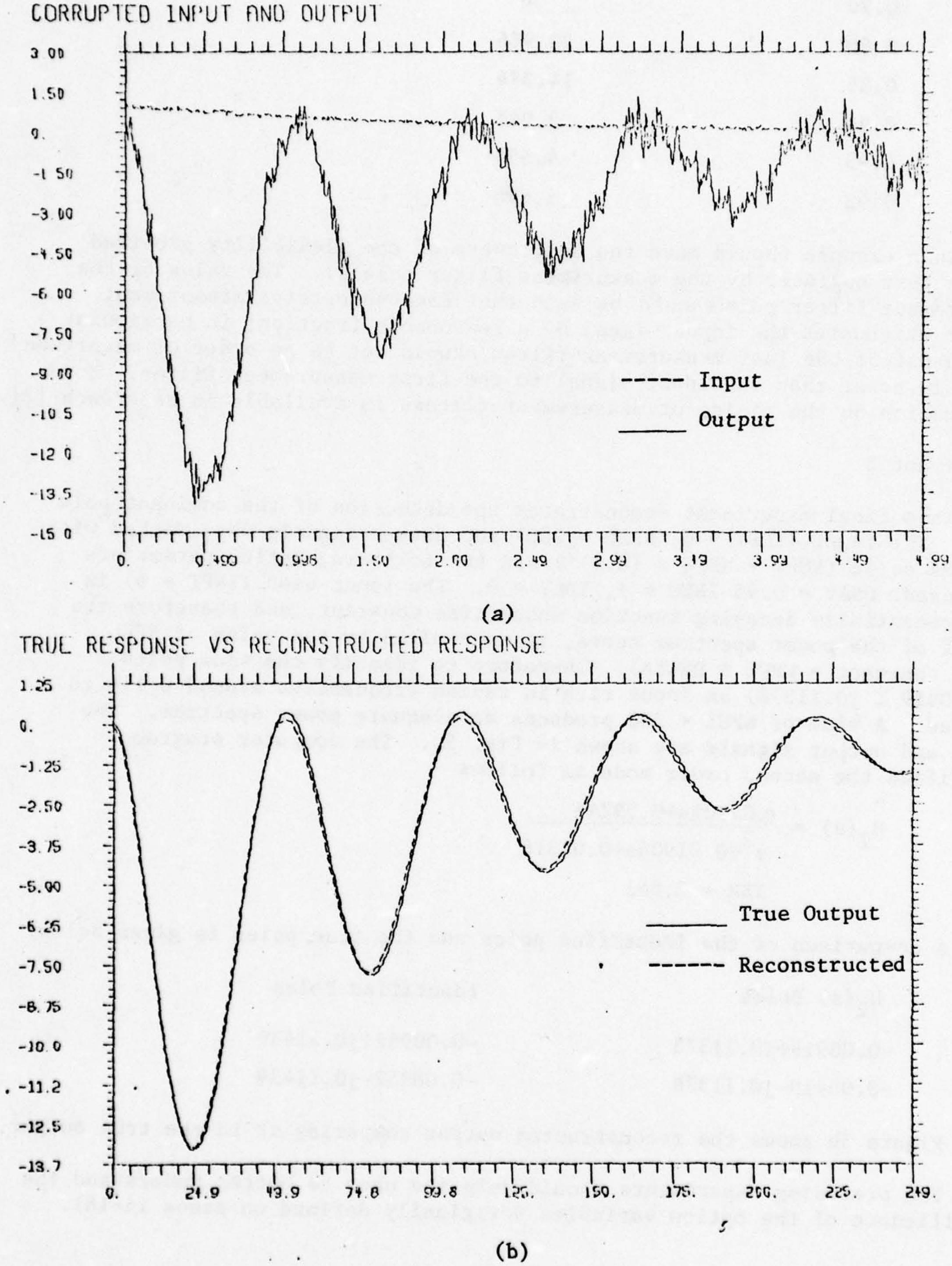
$$ESR = 2.641$$

A comparison of the identified poles and the true poles is given below.

$H_2(s)$ Poles	Identified Poles
-0.00919+j0.11378	-0.00952+j0.11439
-0.00919-j0.11378	-0.00952-j0.11439

Figure 5b shows the reconstructed output comparing it to the true output.

The preceding experiments should help the user to better understand the significance of the option variables (originally defined on pages 14-18).



(a)

(b)

Figure 5. Experiment 5 - Detection of Dominant Complex Poles.

Example 2

Flight tests on a towed-sonar vehicle (SMS2619--- one-third scale model), designed by the Naval Coastal Systems Laboratory, were conducted at the Naval Ship Research and Development Center, Carderock, Md. The test data were recorded on magnetic tapes at a sampling rate of 30Hz. In this example the results of two experiments are presented, one pertaining to pitch vs. stern-plane deflection and the other pertaining to roll vs. rudder deflection. In both cases the data were preprocessed by a 2Hz digital filter to substantially remove an undesirable 3.3Hz oscillation, suspected to be caused by an artifact in instrumentation. The data were then sampled at 5Hz (Nyquist frequency = 2.5Hz ≡ 15.71 radian/sec) for use by the identification program.

For the pitch vs. stern-plane data a second order (N=2) identification was performed which yielded the model

$$\hat{H}_2 = \theta(s)/\delta_s(s) = \frac{0.784(s + 0.028)}{(s + 1.21)(s + 0.03)}$$

The reconstructed output is shown in Fig. 6 together with the output data used for identification. It is worth mentioning that higher order models were also attempted, however, the additional poles found had insignificant energies associated with them.

For the roll vs. rudder data the results of a sixth order (N = 6) identification are presented. The model found is

$$H_6(s) = \frac{-0.09682s^5+1.01434s^4+0.19137s^3+0.04870s^2+0.00132s+0.00013}{s^6+0.67744s^5+0.64971s^4+0.12743s^3+0.02647s^2+0.00078s+0.00007}$$

$$= \frac{-0.35521 \pm j0.79885}{s+0.23747 \pm j0.66789} + \frac{-0.00106 \pm j0.00045}{s+0.01006 \pm j0.05316} + \frac{-0.01183 \pm 0.01648}{s+0.09119 \pm j0.19015}$$

The reconstructed output is shown in Fig. 7 together with the output data. However, the energies associated with the last two pairs of poles are quite small so that a second order identification would therefore seem desirable. In our analysis of the SMS vehicle we also conducted multiinput-multioutput identification, and such analysis showed that the lateral dynamics is essentially governed by two pole-pairs,one pole pair reasonably observable in the roll data and the other in yaw-rate data. The multiinput, multioutput version of GRAM identifier is discussed in [9].

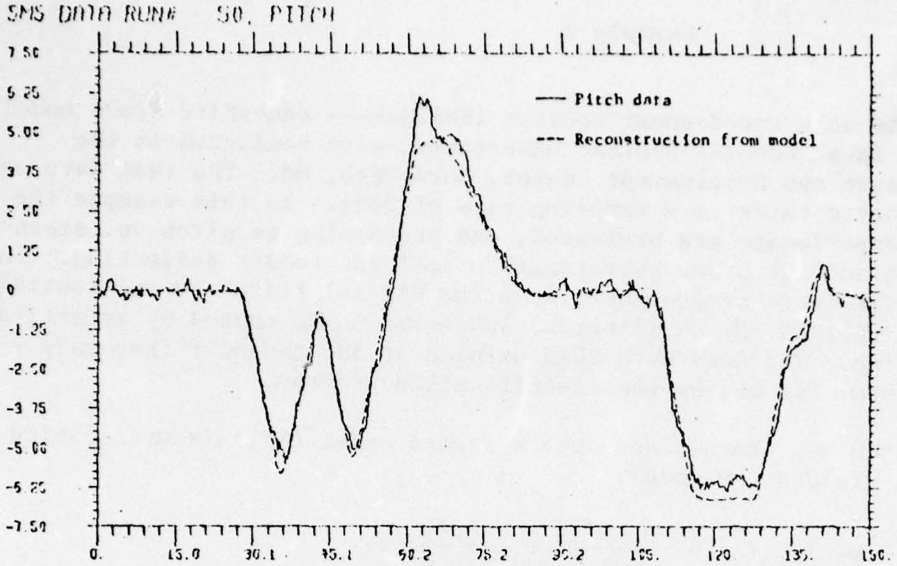


Fig. 6 Identification of longitudinal dynamics of SMS2619 vehicle; reconstruction of pitch from identified model.

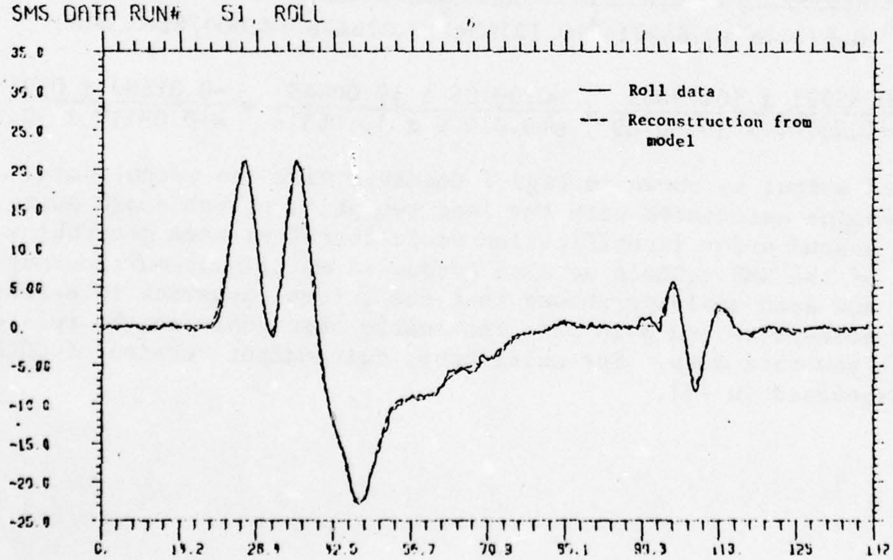


Fig. 7. Identification of lateral dynamics of SMS2619 vehicle; reconstruction of roll from identified model.

REFERENCES

[1] D. E. Humphreys, "Development of the Equations of Motion and Transfer Functions for Underwater Vehicles", Naval Coastal Systems Laboratory Report TR-287-76, July 1976.

[2] G. J. Dobeck, System Identification and Application to Undersea Vehicles, Ph.D. Dissertation, College of Engineering, University of South Florida, 1976.

[3] V. K. Jain, "Extraction of Vehicle Transfer Functions from Noisy Flight Test Data via a Discrete Decoupled Technique (Phase I)", Engineering Research Report, University of South Florida, (submitted to NCSL in November 1974), 1974.

[4] V. K. Jain, "Extraction of Vehicle Transfer Functions from Noisy Flight Test Data via a Discrete Decoupled Technique (Phase II)", Engineering Research Report, University of South Florida, (submitted to NCSL in August 1975), 1975.

[5] H. Freeman, Discrete Time Systems, New York; John Wiley, 1965.

[6] M. J. Levin, "Estimation of a System Pulse Transfer Function in the Presence of Noise", IEEE Trans. A.C., vol. AC-9, 229-235, July, 1964.

[7] G. J. Dobeck, System and Signal Identification by Digital Methods, M.S. Thesis, University of South Florida, Tampa, Florida, 1973.

[8] M. Nichols, "On the z-domain Description of Navy Vehicles", correspondence to Naval Coastal System Laboratory, Panama City, FL, July 1976.

[9] V. K. Jain and G. J. Dobeck, "Multiinput-Multioutput Identification of Vehicle Dynamics via Gram Identifies", Research report in Preparation.

APPENDIX A

SUBROUTINES USED IN PROGRAM

The following subroutines used by the GRAM Identifier program are described in the appendix.

BUILDA
BUILDZ
CONVOL
CORUPT
ERROR
FILLV
FINDQ
GRAMII
IZTOS
POLCON
PRCVEC
PRMAT
PRVEC
RESPON
SOLVE1
SOLVE2
SOLVE3
ZTOS

The subroutines DOBINV, PLOP8 and POLRT are not detailed. Their function is indicated below and they can be substituted by standard routines from a scientific package.

DOBINV   Inversion of a square matrix

PLOP8   X-Y plotter (CALCOMP) routine
The subroutines called by PLOP8 are also not discussed here

POLRT   Computes the real and complex roots of a real polynomial

SUBROUTINE: BUILD A

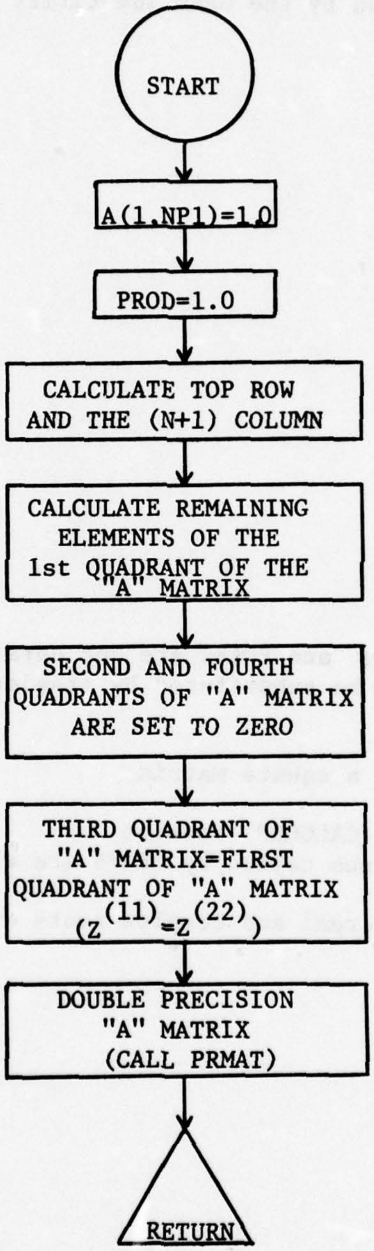
PURPOSE: CALCULATES THE TRANSFORMATION MATRIX TO CONVERT SYNTHETIC PARAMETER VECTOR TO THE DESIRED PARAMETER VECTOR

EQUATIONS: $A_{ij} = A_{i,j+1} - \frac{Q_j A_{i-1,j+1}}{P_j}$ after 1st row and $(n+1)^{th}$ column

"A"'s are generated by other means with $i = 2,3,\ldots., n+1$;

$j=n,n-1,\ldots.,1$

FLOW CHART:


START
A(1,NP1)=1.0
PROD=1.0
CALCULATE TOP ROW AND THE (N+1) COLUMN
CALCULATE REMAINING ELEMENTS OF THE 1st QUADRANT OF THE "A" MATRIX
SECOND AND FOURTH QUADRANTS OF "A" MATRIX ARE SET TO ZERO
THIRD QUADRANT OF "A" MATRIX=FIRST QUADRANT OF "A" MATRIX (Z(11)=Z(22))
DOUBLE PRECISION "A" MATRIX (CALL PRMAT)
RETURN

SUBROUTINE: BUILD A

DESCRIPTION: The "A" Matrix is formed in the following manner

$$A = \begin{bmatrix} \Gamma & 0 \\ & \\ 0 & \Gamma \end{bmatrix}$$

where each quadrant is an (n+1) x (n+1)square matrix.

This matrix embodies the relationship between the synthetic coefficient vector XLAMDA derived from the GRAM matrix and the true coefficient vector of the systems transfer function, GAMMA. This relationship is dependent upon the values used for the measurement filters. Multiplying XLAMDA by a matrix of order 2(n+1) with Γ in the upper left and lower right quadrants, yields GAMMA.

PROGRAM VARIABLES:	A	"A" MATRIX
	DEL	MEASUREMENT FILTER NUMERATOR
	MAX	MAXIMUM ROWS PERMISSIBLE
	N	ORDER OF SYSTEM
	Q	MEASUREMENT FILTER POLE(S)

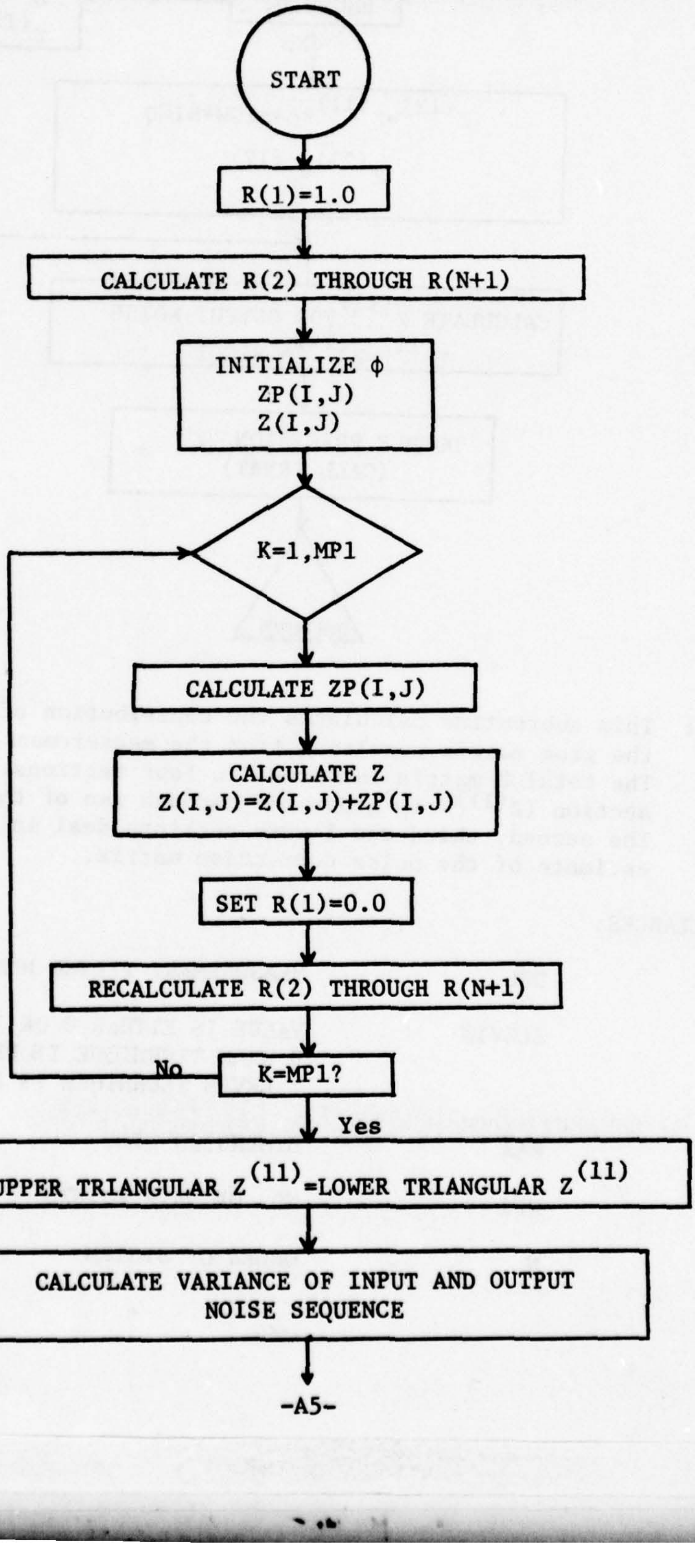
```
      SUBROUTINE BUILDA(A,Q,DEL,N,MAX)
      REAL*8 A(MAX,1),Q(1),DEL(1),PROD
      NP1=N+1
      NPNP2=N+N+2
      A(1,NP1)=1.0D00
      PROD=1.0D00
      DO312K=1,N
      I=NP1-K
      PROD=PROD/DEL(I)
      A(1,I)=PROD
312   A(K+1,NP1)=0.0D00
      DO313I=2,NP1
      DO313K=1,N
      J=NP1-K
313   A(I,J)=(A(I,J+1)-Q(J)*A(I-1,J+1))/DEL(J)
      DO314I=1,NP1
      DO314J=1,NP1
      A(I,J+NP1)=0.0D00
      A(I+NP1,J)=0.0D00
314   A(I+NP1,J+NP1)=A(I,J)
      WRITE(6,1005)
1005  FORMAT(1X,'A-MATRIX')
      CALL PRMAT(A,NPNP2,NPNP2,MAX)
      RETURN
      END
```

SUBROUTINE: BUILD Z

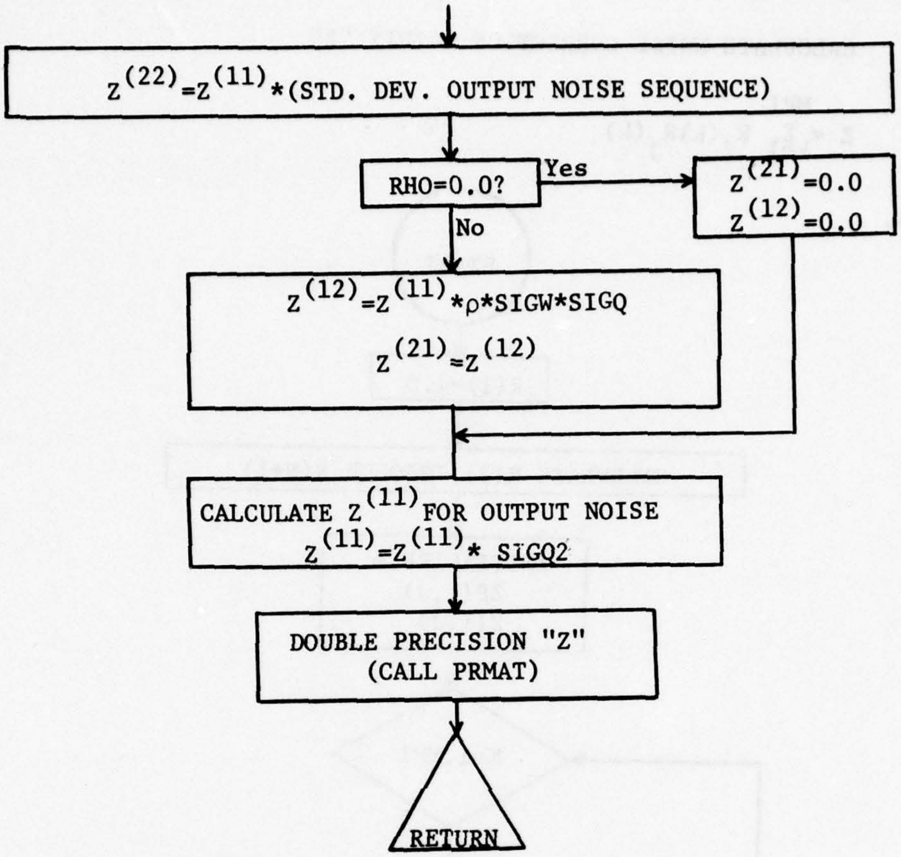
PURPOSE: CALCULATE NOISE CORRECTION MATRIX "Z"

EQUATION: $Z = \sum_{k=1}^{MP1} R_i(k)R_j(k)$

FLOW CHART:


START
R(1)=1.0
CALCULATE R(2) THROUGH R(N+1)
INITIALIZE φ
ZP(I,J)
Z(I,J)
K=1,MP1
CALCULATE ZP(I,J)
CALCULATE Z
Z(I,J)=Z(I,J)+ZP(I,J)
SET R(1)=0.0
RECALCULATE R(2) THROUGH R(N+1)
No
K=MP1?
Yes
UPPER TRIANGULAR Z$^{(11)}$=LOWER TRIANGULAR Z$^{(11)}$
CALCULATE VARIANCE OF INPUT AND OUTPUT
NOISE SEQUENCE

SUBROUTINE: BUILD Z


$Z^{(22)}=Z^{(11)}$*(STD. DEV. OUTPUT NOISE SEQUENCE)
RHO=0.0?
Yes
$Z^{(21)}=0.0$
$Z^{(12)}=0.0$
No
$Z^{(12)}=Z^{(11)}*\rho$*SIGW*SIGQ
$Z^{(21)}=Z^{(12)}$
CALCULATE $Z^{(11)}$ FOR OUTPUT NOISE
$Z^{(11)}=Z^{(11)}$* SIGQ2
DOUBLE PRECISION "Z"
(CALL PRMAT)
RETURN

DESCRIPTION: This subroutine calculates the contribution of the noise to the gram matrix resulting from the measurement filter output. The total Z matrix is formed in four sections. The First section ($Z^{(11)}$) is generated through use of the GAMMA matrix. The second, third and fourth sections deal in optimizing the estimate of the noise correction matrix.

PROGRAM VARIABLES:

DEL	MEASUREMENT FILTER NUMERATOR
ILEVIN	VALUE IS EITHER 0 OR 1. 0 GRAM TECHNIQUE I S PERFORMED 1 LEVEN TECHNIQUE IS PERFORMED
MAX	DIMENSION SIZE
MP1	NO. OF DATA POINTS
N	ORDER OF SYSTEM

SUBROUTINE: **BUILD Z**

Q	MEASUREMENT FILTER POLE
R	COEFFICIENT VECTOR
RHO	EXPECTATION OF (W(K)*Q(K))
SIGQ	STANDARD DEVIATION OF OUTPUT NOISE SEQUENCE
SIGW	STANDARD DEVIATION OF INPUT NOISE SEQUENCE
Z	NOISE CORRECTION MATRIX
ZP	WORKING ARRAY

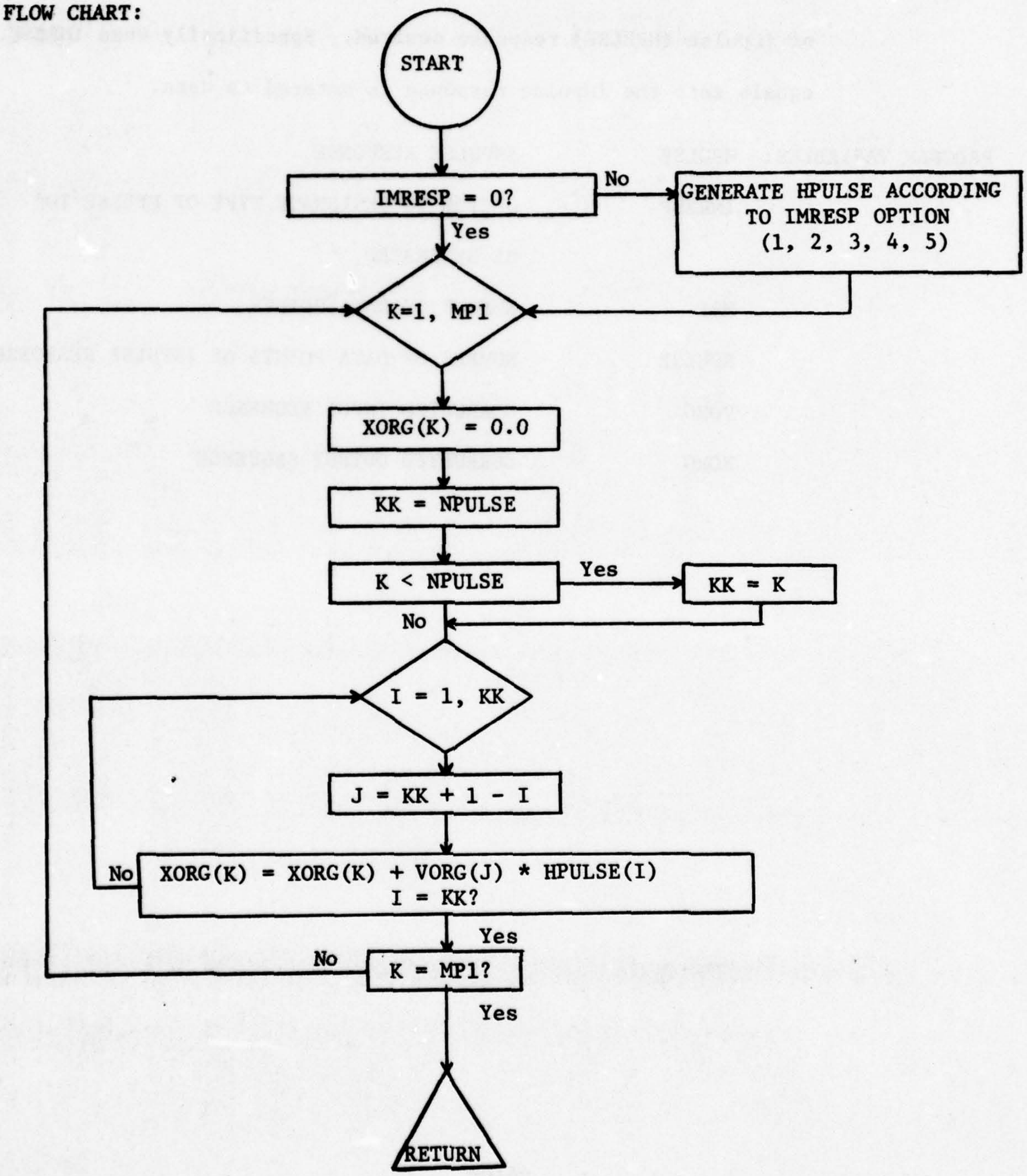
```
      SUBROUTINE BUILDZ(Z,ZP,R,N,MP1,SIGW,SIGQ,RHO,DEL,Q,MAX,ILEVIN)
C
C        SUBROUTINE FOR CALCULATING THE NOISE CORRECTION MATRIX, Z,
C        FOR GRAMI AND GRAMII
C
      DIMENSION Z(MAX,1),ZP(MAX,1),R(1) ,Q(1),DEL(1)
      DOUBLE PRECISION Z,ZP,R,DEL,Q,DCON
      NP1=N+1
      NPNP2=N+N+2
      R(1)=1.0D00
      DO12I=1,N
      R(I+1)=R(I)*DEL(I)
      IF(ILEVIN.EQ.1)R(I+1)=0.0D00
12    CONTINUE
      DO1I=1,NP1
      DO1J=1,NP1
      ZP(I,J)=0.0D00
1     Z(I,J)=0.0D00
      DO2K=1,MP1
      DO3 I=1,NP1
      DO3 J=1,I
      ZP(I,J)=ZP(I,J)+R(I)*R(J)
3     Z(I,J)=Z(I,J)+ZP(I,J)
      IF(ILEVIN.EQ.0)R(1)=0.0D00
      DO4I=1,N
      IF(ILEVIN.EQ.1)R(I+1)=R(I)
      IF(ILEVIN.EQ.1)GO TO 4
      R(I+1)=R(I+1)*Q(I)+R(I)*DEL(I)
4     CONTINUE
2     CONTINUE
      DO40I=1,N
      IP1=I+1
      DO40J=IP1,NP1
40    Z(I,J)=Z(J,I)
      SIGQ2=SIGQ*SIGQ
      SIGW2=SIGW*SIGW
      DO5I=1,NP1
      DO5J=1,NP1
5     Z(NP1+I,NP1+J)=Z(I,J)*SIGW2
      IF(RHO)6,7,6
7     DO8I=1,NP1
      DO8J=1,NP1
      Z(I+NP1,J)=0.0D00
8     Z(I,NP1+J)=0.0D00
      GO TO 9
6     DO11I=1,NP1
      DO11J=1,NP1
      Z(I,NP1+J)=Z(I,J)*RHO*SIGW*SIGQ
11    Z(I+NP1,J)=Z(I,NP1+J)
9     DO10I=1,NP1
      DO10J=1,NP1
10    Z(I,J)=Z(I,J)*SIGQ2
      WRITE(6,1000)
1000  FORMAT(1X,'NOISE CORRECTION MATRIX, Z')
      CALL PRMAT(Z,NPNP2,NPNP2,MAX)


      RETURN
      END
```

SUBROUTINE: CONVOL

PURPOSE: PERFORM CONVOLUTION OF HPULSE AND VORG

EQUATION: $$XORG = \sum_{m=1}^{NPULSE} VORG_{k+1-m} \; HPULSE_{m}$$

FLOW CHART:



SUBROUTINE: CONVOL

DESCRIPTION: This subroutine determines the convolution (XORG(K)) of VORG(I) and HPULSE(J). NPULSE is the length of the impulse response and the variable IMRESP is the option used to specify the type of impulse (HPULSE) response desired. Specifically when IMRESP equals zero the impulse response is entered as data.

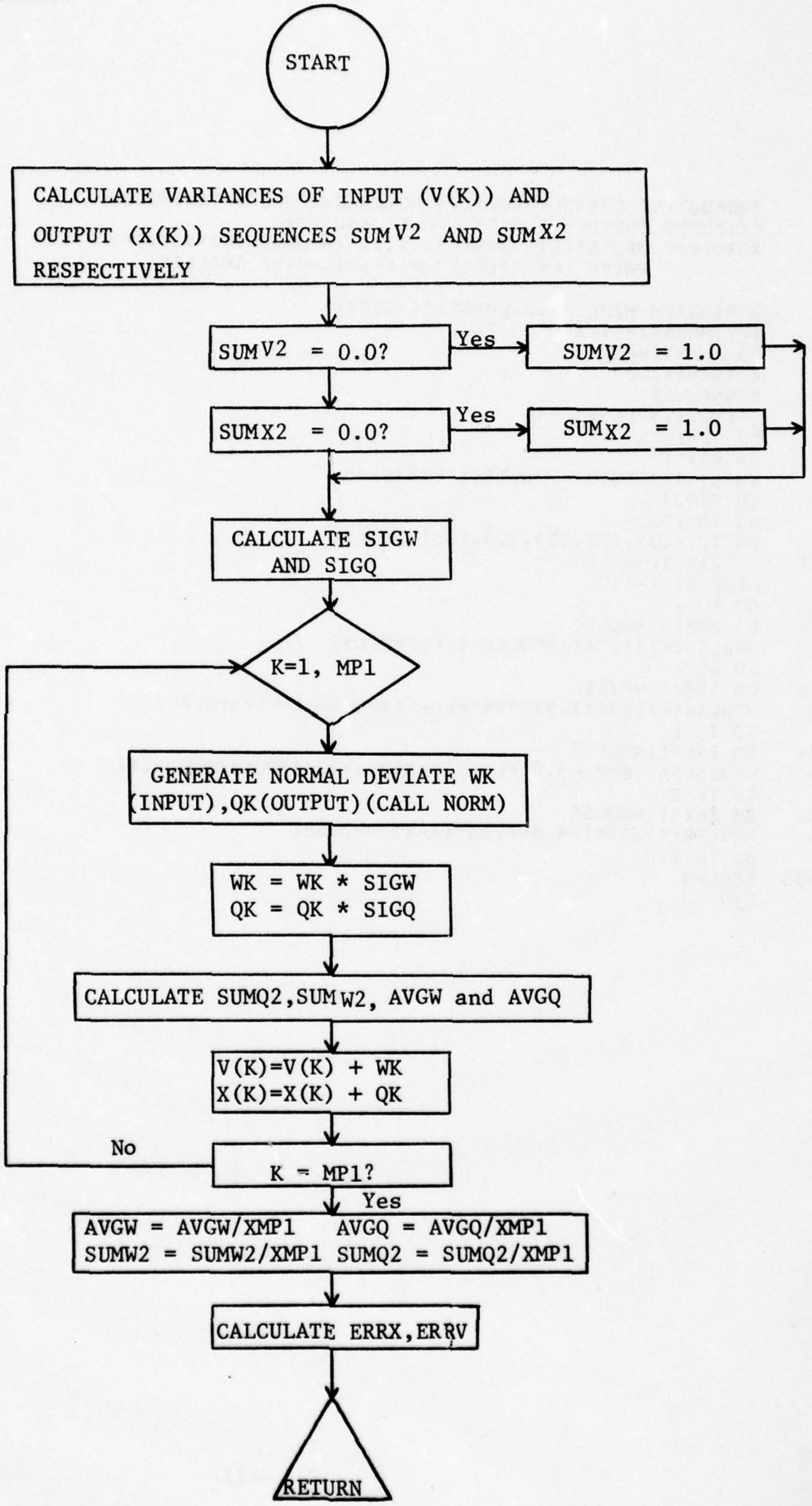
PROGRAM VARIABLES:	HPULSE	IMPULSE RESPONSE
	IMRESP	OPTION TO DESIGNATE TYPE OF HPULSE TO BE GENERATED
	MP1	NUMBER OF DATA POINTS
	NPULSE	NUMBER OF DATA POINTS OF IMPULSE RESPONSE
	VORG	CORRUPTED INPUT SEQUENCE
	XORG	CORRUPTED OUTPUT SEQUENCE

```
      SUBROUTINE CONVOL(HPULSE,VORG,XORG,NPULSE,MP1,IMRESP)
C     PERFORMS CONVOLUTION OF HPULSE AND VORG
C     XORG(K)= HPULSE(1)*VORG(K)+ .... +HPULSE(KK)*VORG(K-KK+1),
C             WHERE  KK=NPULSE OR K WHICHEVER SMALLER
C
      DIMENSION HPULSE(1),VORG(1),XORG(1)
      IF(IMRESP.NE.0)GO TO 20
2     DO 5 K=1,MP1
      XORG(K)=0.0
      KK=NPULSE
      IF(K.LT.NPULSE)KK=K
      DO 4I=1,KK
      J= K+1-I
4     XORG(K)=XORG(K)+HPULSE(I)*VORG(J)
5     CONTINUE
      GO TO 1000
20    GO TO (101,102,103,104,105),IMRESP
101   DO 21K=1,NPULSE
21    HPULSE(K)=1.0
      GO TO 2
102   DO 22K=1,NPULSE
22    HPULSE(K)=FLOAT(NPULSE+1-K)/NPULSE
      GO TO 2
103   DO 23K=1,NPULSE
23    HPULSE(K)=COS(1.570796*FLOAT(NPULSE+1-K)/NPULSE)
      GO TO 2
104   DO 24K=1,NPULSE
24    HPULSE(K)=EXP(-8.0*FLOAT(K*K-2*K+1)/(NPULSE*NPULSE))
      GO TO 2
105   DO 25K=1,NPULSE
25    HPULSE(K)=EXP(-4.0*FLOAT(K-1)/NPULSE)
      GO TO 2
1000  RETURN
      END
```

SUBROUTINE: CORUPT

EQUATION: See attached sheet

FLOW CHART:


START
CALCULATE VARIANCES OF INPUT (V(K)) AND OUTPUT (X(K)) SEQUENCES SUMV2 AND SUMX2 RESPECTIVELY
SUMV2 = 0.0?
Yes
SUMV2 = 1.0
SUMX2 = 0.0?
Yes
SUMX2 = 1.0
CALCULATE SIGW AND SIGQ
K=1, MP1
GENERATE NORMAL DEVIATE WK (INPUT),QK(OUTPUT)(CALL NORM)
WK = WK * SIGW
QK = QK * SIGQ
CALCULATE SUMQ2,SUMW2, AVGW and AVGQ
V(K)=V(K) + WK
X(K)=X(K) + QK
No
K = MP1?
Yes
AVGW = AVGW/XMP1 AVGQ = AVGQ/XMP1
SUMW2 = SUMW2/XMP1 SUMQ2 = SUMQ2/XMP1
CALCULATE ERRX,ERRV
RETURN

SUBROUTINE: CORUPT

EQUATION: $AVGW = \frac{1}{XMP1} \sum_{k=1}^{MP1} WK$ Sample mean of input noise seq.

$AVGQ = \frac{1}{XMP1} \sum_{k=1}^{MP1} QK$ Sample mean of output noise seq.

$SUMQ2 = \frac{1}{XMP1} \sum_{k=1}^{MP1} QK^2$ Sample mean of power of output noise

$SUMW2 = \frac{1}{XMP1} \sum_{k=1}^{MP1} WK^2$ Sample mean power of input noise

$SUMX2 = \frac{1}{XMP1} \sum_{k=1}^{MP1} X(k)^2$ Sample mean power of output seq.

$SUMV2 = \frac{1}{XMP1} \sum_{k=1}^{MP1} V(k)^2$ Sample mean power of input seq.

SUBROUTINE: CORUPT

DESCRIPTION: This subroutine calculates the standard deviation of Input (SIGW) noise sequence and Output(SIGQ) noise sequence. The values of the mean power of input and output noise along with the mean power of input and output sequence are also calculated in this subroutine. The final calculation is of the noise to signal power ratio of both input and output.

PROGRAM VARIABLES:

MP1	NUMBER OF DATA POINTS
SIGQ	STANDARD DEVIATION OF THE OUTPUT NOISE SEQUENCE Q(k)
SIGW	STANDARD DEVIATION OF THE INPUT NOISE SEQUENCE W(k)
V	CORRUPTED INPUT SEQUENCE
X	CORRUPTED OUTPUT SEQUENCE

```
      SUBROUTINE CORUPT(X,V,SIGQ,SIGW,MP1)
      DIMENSION X(1),V(1)
      REAL*8 SUMV2,SUMX2,SUMQ2,SUMW2,AVGQ,AVGW
C
C         IX INITIALIZES UNIFORM RANDOM NUMBER GENERATOR (IBM SUBROUTINE RANDU)
C         RANDU IS CALLED BY SUBROUTINE NORM WHICH GENERATES NORMAL DEVIATES
C
C
C         INITIATE THE RANDOM SEQUENCE GENERATOR
      IX=65549
      XMP1=MP1
      SUMX2=0.0D00
      SUMV2=0.0D00
      DO39K=1,MP1
      SUMX2=SUMX2+X(K)*X(K)
      SUMV2=SUMV2+V(K)*V(K)
39    CONTINUE
      SUMV2=SUMV2/XMP1
      SUMX2=SUMX2/XMP1
      IF(SUMV2.EQ.0.0D0)SUMV2=1.0D0
      IF(SUMX2.EQ.0.0D0)SUMX2=1.0D0
C
C         FOR INPUT=0, SIGW AND SIGQ BECOME STD. DEV. OF NOISE
C
      SIGW=DSQRT(SUMV2*SIGW)
      SIGQ=DSQRT(SUMX2*SIGQ)
      WRITE(6,897)SIGQ,SIGW
897   FORMAT(//30X,'SIGQ=',G17.10,5X,'SIGW=',G17.10,//)
      SUMQ2=0.0D00
      SUMW2=0.0D00
      AVGQ=0.0D00
      AVGW=0.0D00
      DO40K=1,MP1
      CALL NORM(WK,IX)
      CALL NORM(QK,IX)
      WK=WK*SIGW
      QK=QK*SIGQ
      SUMQ2=SUMQ2+QK*QK
      SUMW2=SUMW2+WK*WK
      AVGW=AVGW+WK
      AVGQ=AVGQ+QK
      V(K)=V(K)+WK
40    X(K)=X(K)+QK
      AVGW=AVGW/XMP1
      AVGQ=AVGQ/XMP1
      SUMW2=SUMW2/XMP1
      SUMQ2=SUMQ2/XMP1
      ERRX=DSQRT(SUMQ2/SUMX2)*100.0
      ERRV=DSQRT(SUMW2/SUMV2)*100.0
      WRITE(6,1001)AVGQ,AVGW,SUMQ2,SUMW2,SUMX2,SUMV2,ERRX,ERRV
1001  FORMAT(///,20X,'SAMPLE MEAN OF OUTPUT NOISE SEQUENCE = ',E11.4,/
     1,20X,'SAMPLE MEAN OF INPUT  NOISE SEQUENCE = ',E11.4,/,20X,
     2'SAMPLE MEAN POWER OF OUTPUT NOISE      = ',E11.4,/,20X,
     3'SAMPLE MEAN POWER OF INPUT  NOISE      = ',E11.4,/,20X,
     4'SAMPLE MEAN POWER OF OUTPUT SEQUENCE = ',E11.4,/,20X,


     5'SAMPLE MEAN POWER OF INPUT  SEQUENCE = ',E11.4,/,20X,
     6'100.0 TIMES THE SQUARE ROOT OF THE NOISE TO SIGNAL POWER RATIO OF
     7 THE OUTPUT = ',F7.3,/,20X,
     8'100.0 TIMES THE SQUARE ROOT OF THE NOISE TO SIGNAL POWER RATIO OF
     9 THE INPUT  = ',F7.3)
      RETURN
      END
```

SUBROUTINE: ERROR

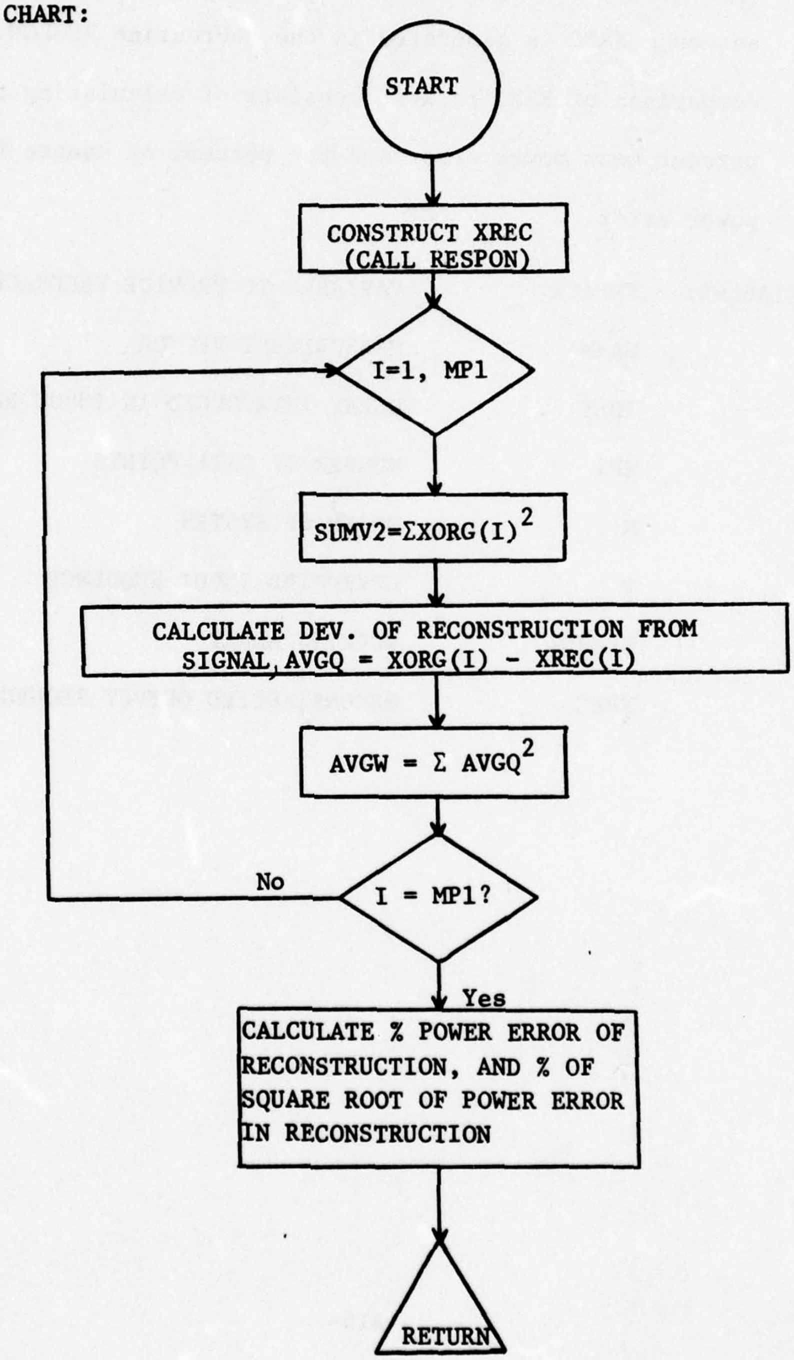
PURPOSE: CALCULATE PERCENT MEAN POWER ERROR IN RECONSTRUCTION AND PERCENT OF SQUARE ROOT OF POWER ERROR IN RECONSTRUCTION.

EQUATIONS:

$$AVGW = \Sigma \left[\frac{(XORG(I) - XREC(I))}{XORG(I)}\right]^2 * 100$$

$$AVGQ = \sqrt{AVGW} * 100$$

FLOW CHART:



SUBROUTINE: ERROR

DESCRIPTION: The subroutine ERROR calculates the Output sequence from the Input sequence and the GAMMA Matrix. The sequence called XREC is compared to XORG, the original output sequence. The sequence XREC is generated in the subroutine RESPON. The comparison of XREC to XORG consists of calculating the percent mean power error and the percent of square root of power error.

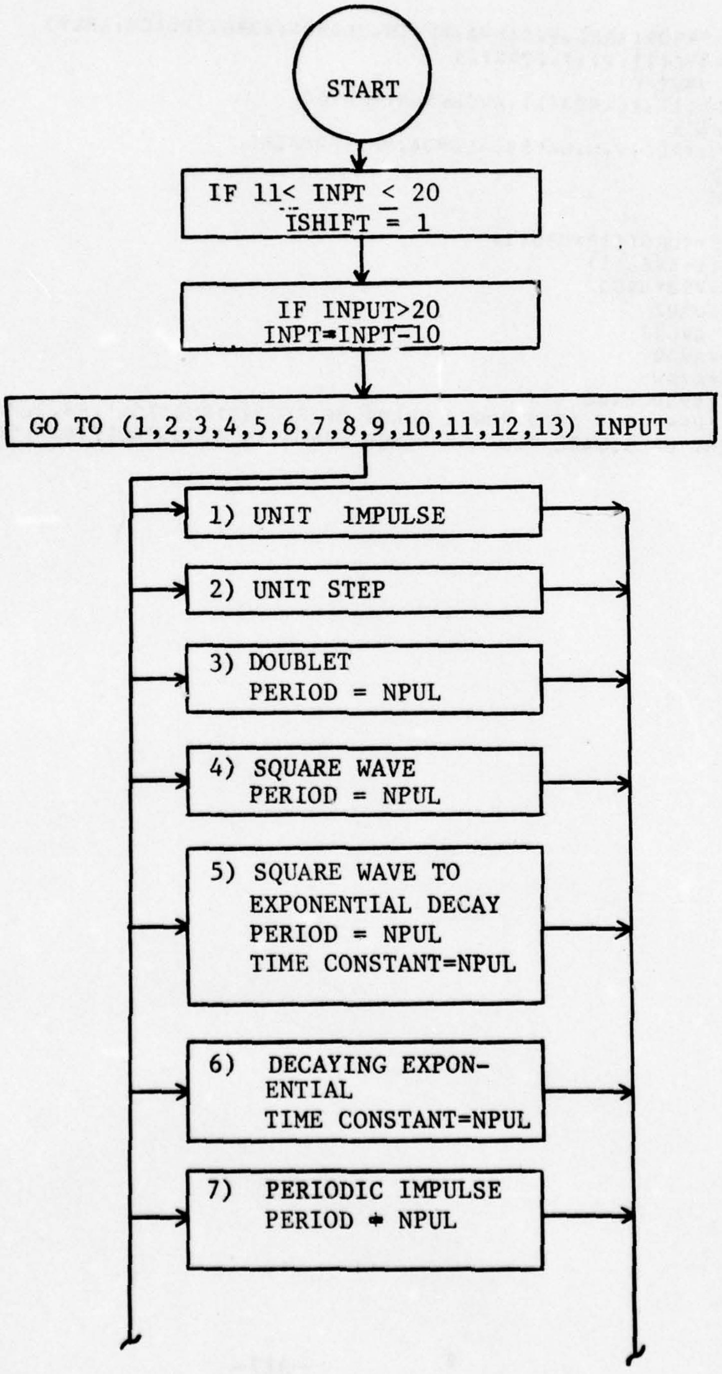
PROGRAM VARIABLES:	FDBACK	VARIABLE TO PROVIDE FEEDBACK IF DESIRED
	GAMMA	MEASUREMENT VECTOR
	IDLY	DELAY INTRODUCED IN INPUT NUMERATOR
	MP1	NUMBER OF DATA POINTS
	N	ORDER OF SYSTEM
	V	CORRUPTED INPUT SEQUENCE
	XLAMDA	WORKING ARRAY
	XREC	RECONSTRUCTED OUTPUT SEQUENCE

```
      SUBROUTINE ERROR(XREC,V,GAMMA,MP1,N,XLAMDA,XORG,FDBACK,IDLY)
      DIMENSION XREC(1),V(1),XORG(1)
      DIMENSION  VVV(20)
      REAL*8 GAMMA(1),XLAMDA(1),AVGW,SUMV2,AVGQ
      INTEGER FDBACK
      CALL RESPON(XREC,V,N,GAMMA,XLAMDA,MP1,FDBACK)
      AVGW=0.0D00
      SUMV2=0.0D0
      DO26I=1,MP1
      SUMV2=SUMV2+XORG(I)*XORG(I)
      AVGQ=XORG(I)-XREC(I)
26    AVGW=AVGW+AVGQ*AVGQ
      AVGW=AVGW/SUMV2
      AVGQ=DSQRT(AVGW)
      AVGQ=100.0*AVGQ
      AVGW=100.0*AVGW
      WRITE(6,27)AVGW,AVGQ
27    FORMAT(1X,'PER CENT MEAN POWER ERROR OF RECONSTRUCTION',F8.3,///,
     11X,'PER CENT OF SQUARE ROOT OF POWER ERROR IN RECOSTRUCTION',F8.3)
      RETURN
      END
```

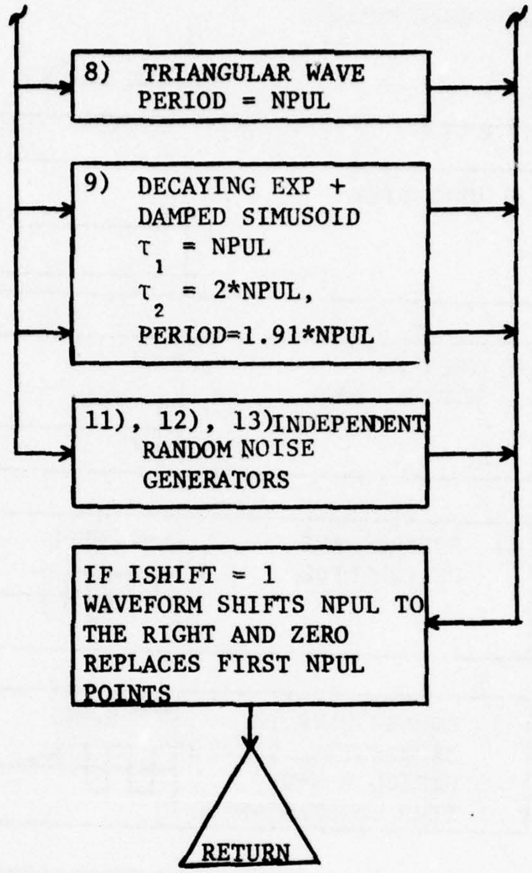
SUBROUTINE: FILLV

PURPOSE: GENERATION OF DISCRETE POINTS FOR A VARIETY OF WAVEFORMS (For the correspondance of the waveshapes and input parameter INPT see page)

FLOW CHART:



SUBROUTINE: FILLV



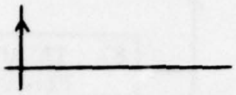
DESCRIPTION: This subroutine builds an array of NPT points defined by the choosen wave form and parameter (NPUL) of that waveform. It is useful in approximating input signals for excitation of control system.

PROGRAM VARIABLES:

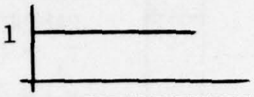
INPUT	DESIRED WAVEFORM OPTION
NPT	NUMBER OF DATA POINTS
NPUL	WAVEFORM PARAMETER
V	GENERATED OUTPUT SEQUENCE

SUBROUTINE: FILLV

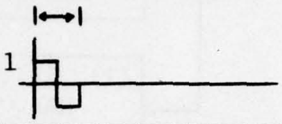
1) UNIT PULSE



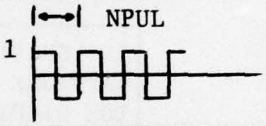
2) UNIT STEP



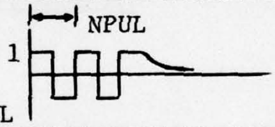
3) DOUBLET
WIDTH = NPUL



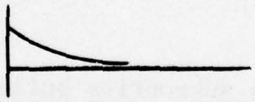
4) SQUARE WAVE
PERIOD=NPUL



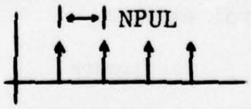
5) SQUARE WAVE TO
EXPONENTIAL DECAY
PERIOD = NPUL
TIME CONSTANT=NPUL



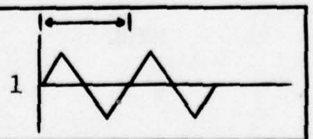
6) EXPONENTIAL
TIME CONSTANT=NPUL



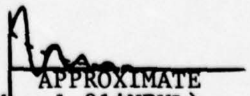
7) PERIODIC IMPULSE
PERIOD=NPUL



8) TRIANGULAR WAVE
PERIOD=NPUL



9) DECAYING EXP +
DAMPED SINUSOID
τ_1 = NPUL
(τ_2 = 2*NPUL, Period = 1.91*NPUL)



```
      SUBROUTINE FILLV(V,NPT,INPUT,NPUL)
C     FILLS THE ARRAY FOR INPUT ACCORDING TO INPUT OPTION DESIGNATED
      DIMENSION V(1)
      GO TO (1,2,3,4,5,6,7,8,9,10),INPUT
1     V(1)=1.0                                                                IMPULSE
      DO 101 I=2,NPT
101   V(I)=0.0
      GO TO 999
2     DO 102 I=1,NPT                                                          STEP
102   V(I)=1.0
      GO TO 999
3     DO 103 I=1,NPT                                                          DOUBLET
      V(I)=0.0
      IF(I.LT.NPUL/2)V(I)=1.
      IF(I.GE.NPUL.AND.I.LT.  NPUL) V(I)=-1.0
103   CONTINUE
      GO TO 999
4     DO 104 I=1,NPT                                                          SQ WAVE
      V(I)=1.0
      TPUL=I/NPUL
      IF(I-TPUL*NPUL.GE.NPUL/2) V(I)=-1.0
104   CONTINUE
      GO TO 999
5     DO 105 I=1,NPT                                                          SQ-EXP
      V(I)=1.0
      IF(I.GE.NPUL/2.AND.I.LT.NPUL) V(I)=-1.0
      IF(I.GE.(1.5)*NPUL.AND.I.LT.2*NPUL) V(I)=-1.0
      ARG1=2.5-FLOAT(I)/FLOAT(NPUL)
      IF(I.GE.(2.5)*NPUL) V(I)=EXP(ARG1)
105   CONTINUE
      GO TO 999
6     DO 106 I=1,NPT                                                          EXP
      ARG2=-FLOAT(I)/FLOAT(NPUL)
106   V(I)=EXP(ARG2)
      GO TO 999
7     DO 107 I=1,NPT                                                          PRD IMPL
107   V(I)=0.0
      N=NPT/NPUL
      V(1)=1.0
      DO 1071 J=1,N
      I=J*NPUL
1071  V(I)=1.0
      GO TO 999
8     DO 108 I=1,NPT                                                          TRI WAVE
      TPUL=I/NPUL
      ITPUL=TPUL
      V(I)=(2.*FLOAT(I)/FLOAT(NPUL)-2.*TPUL)*(-1.0)**ITPUL
      IF(I-TPUL*NPUL.GE.NPUL/2) V(I)=2*(1+TPUL-FLOAT(I)/FLOAT(NPUL))*(-1
     1.0)**ITPUL
108   CONTINUE
      GO TO 999
9     DO 109 I=1,NPT                                                          EXP+OSC
      ARG3=-FLOAT(I)/FLOAT(NPUL)
      ARG4=-.5*FLOAT(I)/FLOAT(NPUL)
      ARG5=3.296*FLOAT(I)/FLOAT(NPUL)
```

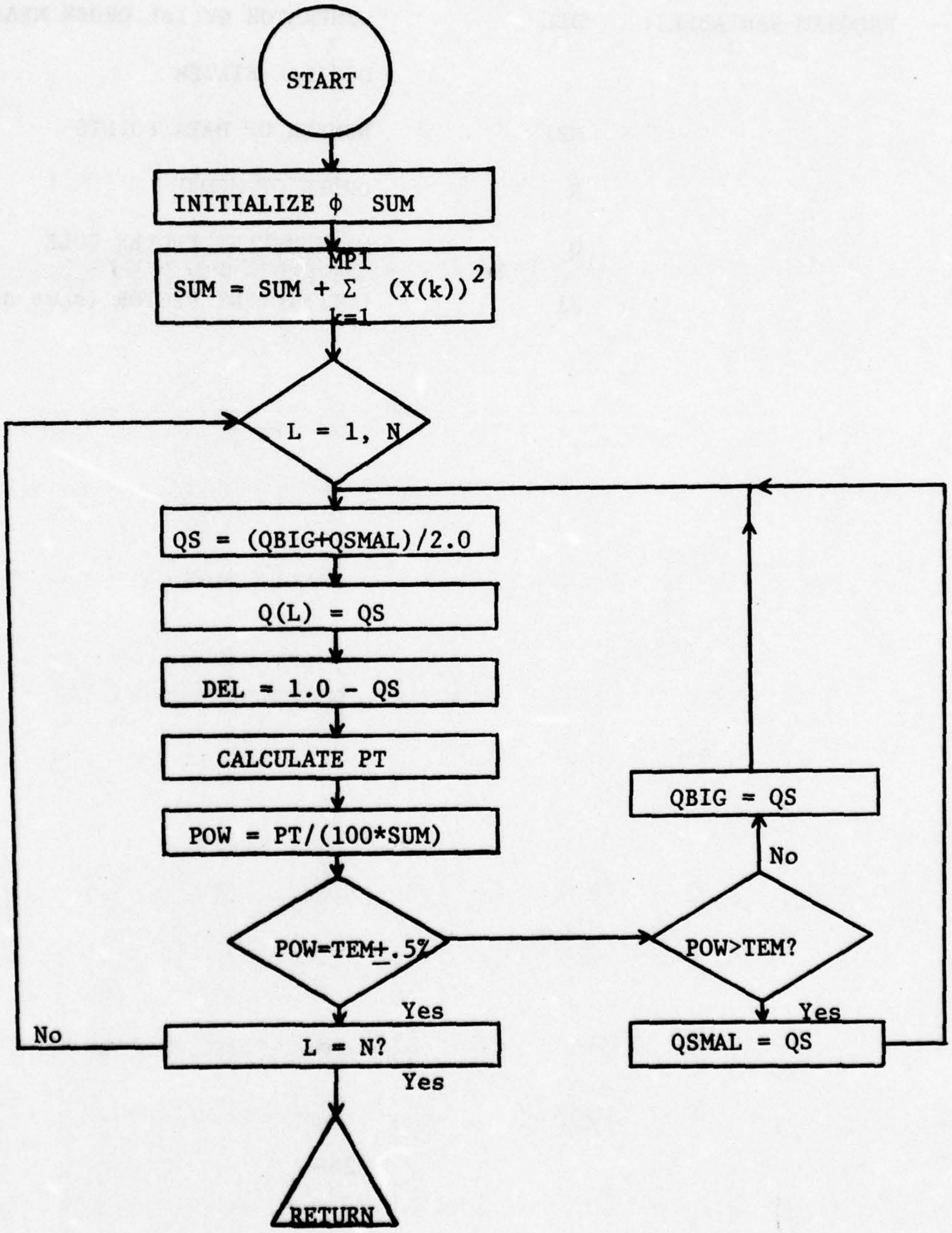
```
109     V(I)=EXP(ARG3)+EXP(ARG4)*SIN(ARG5)
        GO TO 999
10      IX=619027213                                              RANDOM
        DO 110 I=1,NPT
        A=0.0
        DO 1101 K=1,12
        IY=IX*65539
        IF(IY)1102,1103,1103
1102    IY=IY+2147483647+1
1103    YFL=IY
        YFL=YFL*.4656613E-9
        IX=IY
        A=A+YFL
1101    CONTINUE
        V(I)=A-6.0
110     CONTINUE
        GO TO 999
999     CONTINUE
        RETURN
        END
```

SUBROUTINE: FINDQ

PURPOSE: CALCULATES MEASUREMENT FILTER POLE(Q) AND NUMERATOR (DEL).

EQUATION: $H_m(Z) = \frac{DEL}{1-QZ^{-1}}$

FLOW CHART:



SUBROUTINE: FINDQ

DESCRIPTION: FINDQ determines the measurement filter pole and numerator. The subroutine uses an iterative process of calculating DEL and Q. The iteration is satisfied when the variable POW is within ± .5% TEM.

PROGRAM VARIABLES:

DEL	NUMERATOR OF 1st ORDER MEASUREMENT DIGITAL FILTER
MP1	NUMBER OF DATA POINTS
N	ORDER OF MODEL
Q	MEASUREMENT FILTER POLE
X1	COEFFICIENT VECTOR (same as Gamma)

```
      SUBROUTINE FINDQ(Q,DEL,X1,X,MP1,N)
      DIMENSION X(1)
      REAL*8 Q(1),DEL(1),TEM,POW,PT,QS,QBIG,QSMAL,X1(1),SUM
      SUM=0.0D0
      DO667K=1,MP1
667   SUM=SUM+X(K)**2
      NP1=N+1
      DO1L=1,N
      LP1=L+1
      TEM=100.0D0/DFLOAT(NP1)*DFLOAT(NP1-L)
      QBIG=1.0D0
      QSMAL=0.0D0
100   QS=(QBIG+QSMAL)/2.0D0
      Q(L)=QS
      DEL(L)=1.0D0-QS
      PT=0.0D0
      DO4I=1,LP1
4     X1(I)=0.0D0
      DO3K=1,MP1
      X1(1)=X(K)
      DO5I=1,L
5     X1(I+1)=X1(I+1)*Q(I)+X1(I)*DEL(I)
3     PT=PT+X1(LP1)*X1(LP1)
      POW=PT/SUM*100.0D0
      IF(POW.LE.1.005D0*TEM.AND.POW.GE.TEM*0.995D0)GO TO 1
      IF(POW.GT.TEM)GO TO 6
      QBIG=QS
      GO TO 100
6     QSMAL=QS
      GO TO 100
1     CONTINUE
      RETURN
      END
```

SUBROUTINE: GRAMII

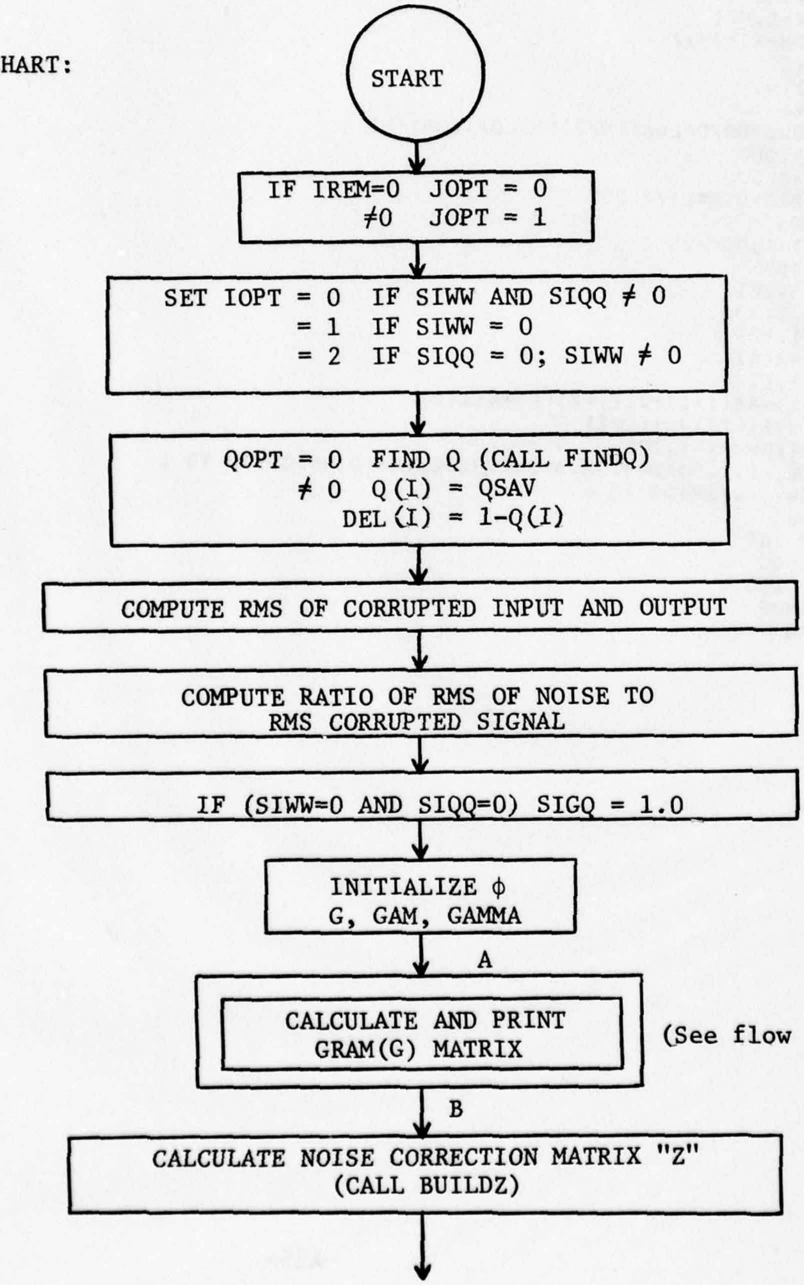
PURPOSE: PERFORMS GRAM II TECHNIQUE WHICH YIELDS THE GRAM MATRIX (G), NOISE CORRECTION MATRIX (Z) AND THE COEFFICIENT MATRIX (GAMMA).

EQUATION:

$$G_{ij} = \sum_{k=1}^{MP1} GAMMA_i(k)*GAMMA_j(k)$$

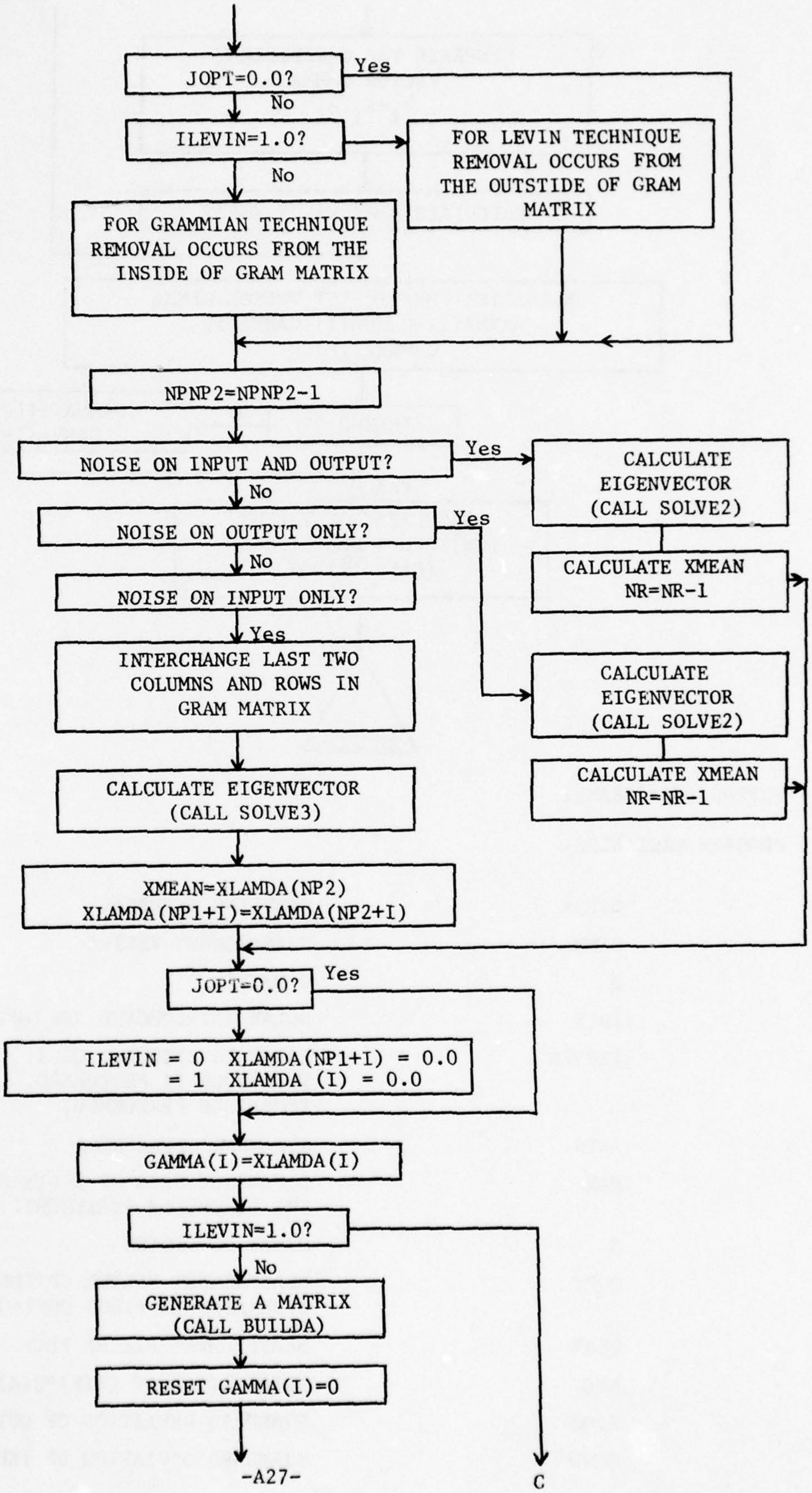
$$\gamma_i = A_{ij}\lambda_j$$

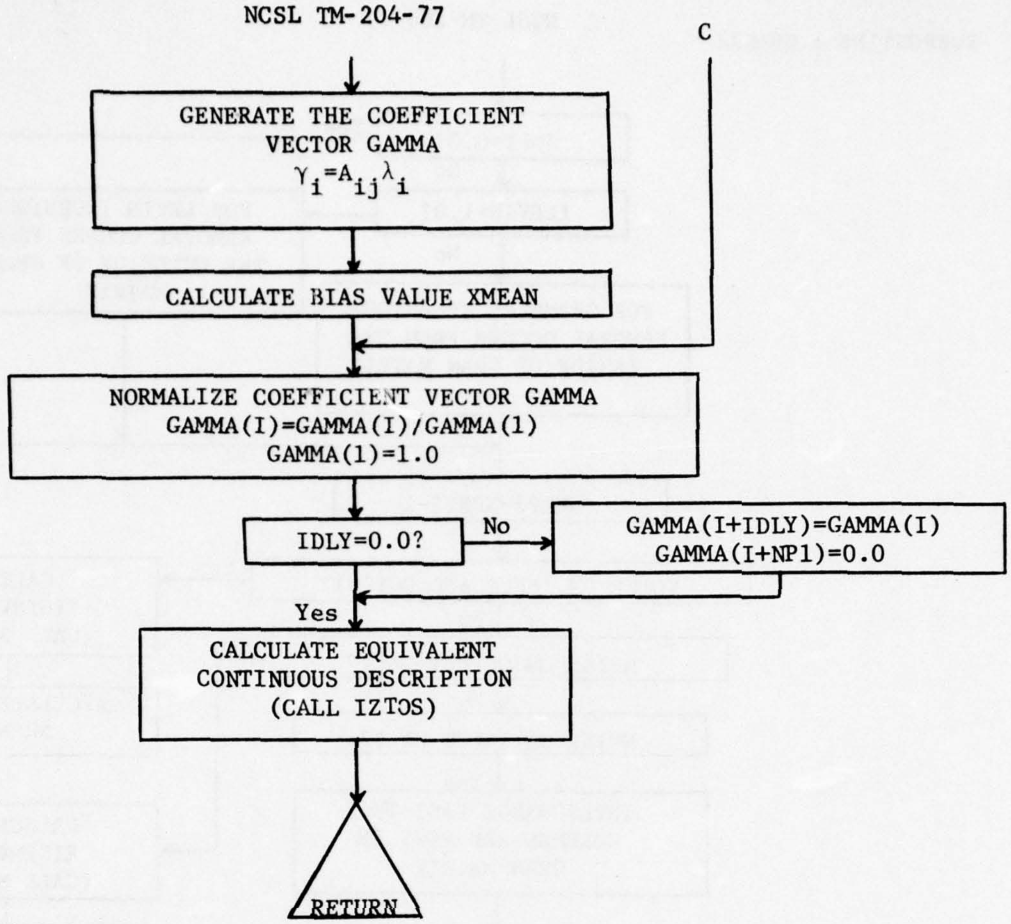
FLOW CHART:


START
IF IREM=0 JOPT = 0
≠0 JOPT = 1
SET IOPT = 0 IF SIWW AND SIQQ ≠ 0
= 1 IF SIWW = 0
= 2 IF SIQQ = 0; SIWW ≠ 0
QOPT = 0 FIND Q (CALL FINDQ)
≠ 0 Q(I) = QSAV
DEL (I) = 1-Q(I)
COMPUTE RMS OF CORRUPTED INPUT AND OUTPUT
COMPUTE RATIO OF RMS OF NOISE TO RMS CORRUPTED SIGNAL
IF (SIWW=0 AND SIQQ=0) SIGQ = 1.0
INITIALIZE ϕ
G, GAM, GAMMA
A
CALCULATE AND PRINT GRAM(G) MATRIX
B
CALCULATE NOISE CORRECTION MATRIX "Z" (CALL BUILDZ)

(See flow chart on page)

SUBROUTINE : GRAMII


JOPT=0.0?
Yes
No
ILEVIN=1.0?
FOR LEVIN TECHNIQUE REMOVAL OCCURS FROM THE OUTSTIDE OF GRAM MATRIX
No
FOR GRAMMIAN TECHNIQUE REMOVAL OCCURS FROM THE INSIDE OF GRAM MATRIX
NPNP2=NPNP2-1
NOISE ON INPUT AND OUTPUT?
Yes
CALCULATE EIGENVECTOR (CALL SOLVE2)
CALCULATE XMEAN NR=NR-1
No
NOISE ON OUTPUT ONLY?
Yes
No
NOISE ON INPUT ONLY?
Yes
INTERCHANGE LAST TWO COLUMNS AND ROWS IN GRAM MATRIX
CALCULATE EIGENVECTOR (CALL SOLVE2)
CALCULATE XMEAN NR=NR-1
CALCULATE EIGENVECTOR (CALL SOLVE3)
XMEAN=XLAMDA(NP2)
XLAMDA(NP1+I)=XLAMDA(NP2+I)
JOPT=0.0?
Yes
ILEVIN = 0 XLAMDA(NP1+I) = 0.0
= 1 XLAMDA (I) = 0.0
GAMMA(I)=XLAMDA(I)
ILEVIN=1.0?
No
GENERATE A MATRIX (CALL BUILDA)
RESET GAMMA(I)=0
C



SUBROUTINE: GRAMII

PROGRAM VARIABLES:

DELTA	SAMPLING INTERVAL
GAMMA	MEASUREMENT VECTOR
G	G MATRIX
IDLY	*DELAY INTRODUCTED ON INPUT NUMERATOR*
ILEVIN	VALUE IS EITHER 0 OR 1. 0 GRAM TECHNIQUE IS PERFORMED. 1 LEVIN TECHNIQUE PERFORMED.
IZTS	SEE SUBROUTINE ZTOS
MAX	DIMENSION SIZE OF 2 DIM ARRAYS IN THE DIMENSION STATEMENT.
N	ORDER OF SYSTEM
QOPT	MEASUREMENT FILTER OPTION. QOPT=1 Q(I) GENERATED IN FINDQ QOPT=0 Q(I)=QSAV
QSAV	MEASUREMENT FILTER POLE
RHO	EXPECTATION OF (W(K)*Q(K))
SIQQ	STANDARD DEVIATION OF OUTPUT NOISE SEQUENCE
SIWW	STANDARD DEVIATION OF INPUT NOISE SEQUENCE

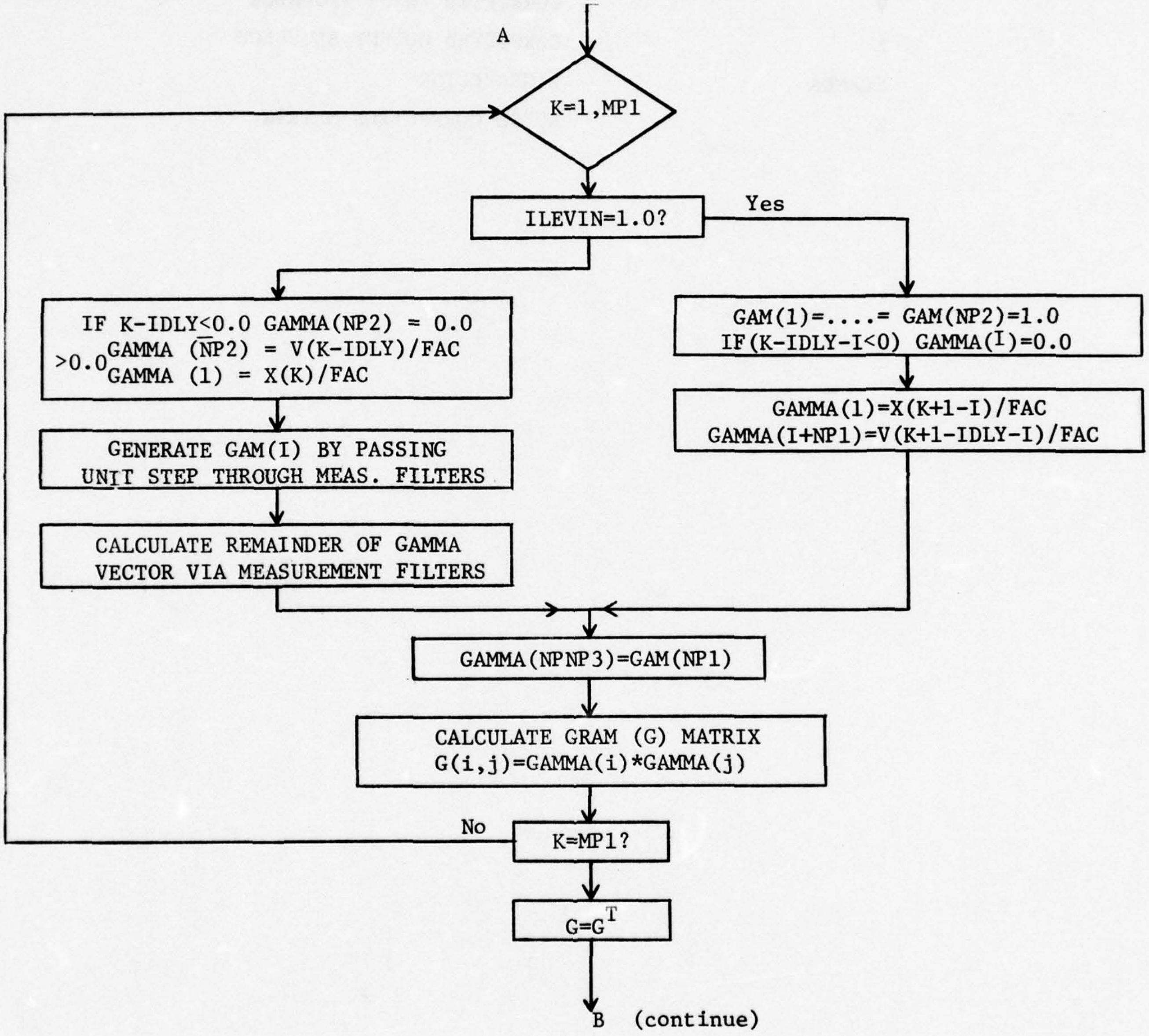
SUBROUTINE: GRAMII

PROGRAM VARIABLES CONTINUED:

V	CORRUPTED INPUT SEQUENCE
X	CORRUPTED OUTPUT SEQUENCE
XLAMDA	EIGENVECTOR
Z	NOISE CORRECTION MATRIX

SUBROUTINE: GRAMII

CALCULATION OF GRAM MATRIX "G"


A
K=1,MP1
ILEVIN=1.0?
Yes
IF K-IDLY<0.0 GAMMA(NP2) = 0.0
>0.0 GAMMA (NP2) = V(K-IDLY)/FAC
GAMMA (1) = X(K)/FAC
GENERATE GAM(I) BY PASSING UNIT STEP THROUGH MEAS. FILTERS
CALCULATE REMAINDER OF GAMMA VECTOR VIA MEASUREMENT FILTERS
GAM(1)=....= GAM(NP2)=1.0
IF(K-IDLY-I<0) GAMMA(I)=0.0
GAMMA(1)=X(K+1-I)/FAC
GAMMA(I+NP1)=V(K+1-IDLY-I)/FAC
GAMMA(NPNP3)=GAM(NP1)
CALCULATE GRAM (G) MATRIX
G(i,j)=GAMMA(i)*GAMMA(j)
No
K=MP1?
G=G^T
B (continue)

```
      SUBROUTINE GRAMII(X,V,MP1,SIQQ,SIWW,RHO,N,DELTA,QSAV,QOPT,IREM,
     1IZTS,GAMMA,XLAMDA,G,Z,MAX,ILEVIN,IDLY,INORM)
C
C     THIS SUBROUTINE PERFORMS THE GRAM II TECHNIQUE
C
      DIMENSION X(1),V(1),G(MAX,1),Z(MAX,1),GAMMA(1),XLAMDA(1),Q(20),
     1DEL(20)
      DOUBLE PRECISION G,Z,GAMMA,XLAMDA,DELTA,DEL,PROD,Q,QSAV
      INTEGER QOPT
      REAL*8 XMEAN,GAM(25),FAC
      REAL*8 S,F,S1,S2,VARQ,VARW,GI
      COMMON /MATRIX/S(20,20),F(20,20),GI(20,20),S1(10,10),S2(10,10)
      MAX2=MAX/2
      WRITE(6,1000)
1000  FORMAT(1H1,20X,'THE GRAM II TECHNIQUE')
C        JOPT = 0 IF DIRECT TRANSMISSION IS ASSUMMED
      JOPT=0
C        JOPT = 1 IF NO DIRECT TRANSMISSION IS ASSUMMED
      IF(IREM.NE.0)JOPT=1
C
C        IOPT = 0 NOISE ON BOTH INPUT AND OUTPUT IS ASSUMED
C        IOPT = 1 NOISE ON OUTPUT ONLY IS ASSUMED
C        IOPT = 2 NOISE ON INPUT ONLY IS ASSUMED
C
      IOPT=0
      IF(SIWW.EQ.0.0)IOPT=1
      IF(SIQQ.EQ.0.0.AND.SIWW.NE.0.0)ICPT=2
C
C        DEL IS THE NUMERATOR OF THE KNOWN FIRST ORDER DIGITAL FILTERS
      IF(QOPT.NE.0) GO TO 21
      DO19I=1,N
      DEL(I)=1.0D00-QSAV
19    Q(I)=QSAV
      GO TO 22
21    CALL FINDQ(Q,DEL,GAMMA,X,MP1,N)
22    CONTINUE
      WRITE(6,2020)
2020  FORMAT(30X,'Q PARAMETERS')
      CALL PRVEC(Q,N)
      NP1=N+1
      NP2=N+2
      NPNP2=N+N+2
      NR=NP1-IREM
      NP1PIR=NP1+IREM
      VARW=0.0
      VARQ=0.0
      DO300I=1,MP1
      VARW=VARW+V(I)*V(I)
300   VARQ=VARQ+X(I)*X(I)
      VARQ=DSQRT(VARQ/MP1)
      VARW=DSQRT(VARW/MP1)
      SIGQ=SIQQ/VARQ
      SIGW=SIWW/VARW
      IF(SIWW.EQ.0.0.AND.SIQQ.EQ.0.0)SIGQ=1.0
      NPNP2=NPNP2+1
      NR=NR+1
      DO10I=1,NPNP2
      GAM(I)=0.0D0
      GAMMA(I)=0.0D00
      DO10J=I,NPNP2
10    G(I,J)=0.0D00
      GAM(1)=1.0D0
C
C         CALCULATING THE G MATRIX
      DO50K=1,MP1
      IF(ILEVIN.EQ.1)GO TO 31
      IF(K-IDLY)25,25,24
25    GAMMA(NP2)=0.0D00
      GO TO 26
24    FAC=1.0
      IF(INORM.EQ.1)FAC=VARW
      GAMMA(NP2)=V(K-IDLY)/FAC
      FAC=1.0
      IF(INORM.EQ.1)FAC=VARQ
      GAMMA(1)=X(K)/FAC
26    CONTINUE
      DO30I=1,N
      GAM(I+1)=GAM(I+1)*Q(I)+GAM(I)*DEL(I)
      GAMMA(I+1)=GAMMA(I)*DEL(I)+GAMMA(I+1)*Q(I)
30    GAMMA(I+NP2)=GAMMA(I+NP1)*DEL(I)+GAMMA(I+NP2)*Q(I)
      GO TO 35
31    CONTINUE
      DO 32 I=1,NP1
      GAM(I+1)=GAM(I)
      IF(K-IDLY-I.LT.0) GAMMA(I)=0.0D00
      IF(K-IDLY-I.LT.0) GAMMA(I+NP1)=0.0D00
      IF(K-IDLY-I.LT.0) GO TO 32
      FAC=1.0
      IF(INORM.EQ.1)FAC=VARQ
      GAMMA(1)=X(K+1-I)/FAC
      FAC=1.0
      IF(INORM.EQ.1)FAC=VARW
      GAMMA(I+NP1)=V(K+1-IDLY-I)/FAC
32    CONTINUE
35    CONTINUE
      GAMMA(NPNP2)=GAM(NP1)
      DO40I=1,NPNP2
      DO40J=I,NPNP2
40    G(I,J)=G(I,J)+GAMMA(I)*GAMMA(J)
50    CONTINUE
      DO60I=2,NPNP2
      K=I-1
      DO60J=1,K
60    G(I,J)=G(J,I)
      WRITE(6,1002)
1002  FORMAT(20X,'THE G MATRIX')
      CALL PRMAT(G,NPNP2,NPNP2,MAX)
C
C         CALCULATING THE NOISE CORRECTION MATRIX Z BY SUBROUTINE BUILDZ
      CALL BUILDZ(Z,S,GAMMA,N,MP1,SIGW,SIGQ,RHO,DEL,Q,MAX,ILEVIN)
      IF(JOPT)70,90,70
70    CONTINUE
      IF(ILEVIN.EQ.1)GO TO 81
      DO80J=1,NPNP2
      DO80I=1,NR
      Z(NP1+I,J)=Z(NP1PIR+I,J)
```

```
80     G(NP1+I,J)=G(NP1PIR+I,J)
       NPNP2=NPNP2-IREM
       DO85J=1,NPNP2
       DO85I=1,NR
       Z(J,NP1+I)=Z(J,NP1PIR+I)
85     G(J,NP1+I)=G(J,NP1PIR+I)
       GO TO 90
81     DO 82 J=1,NPNP2
       Z(NP1+NR+1,J)=G(NPNP2,J)
82     G(NP1+NR+1,J)=G(NPNP2,J)
       NPNP2=NPNP2-IREM
       DO83I=1,NPNP2
       Z(I,NP1+NR+1)=Z(I,NP1+NP1+1)
83     G(I,NP1+NR+1)=G(I,NP1+NP1+1)
       CALL PRMAT(G,NPNP2,NPNP2,MAX)
90     NPNP2=NPNP2-1
       IF(IOPT-1)617,605,618
C
C          NOISE ON BOTH INPUT AND OUTPUT
617    CALL SOLVE2(Z,G,GAMMA,XLAMDA,NPNP2,1,MAX)
       XMEAN=XLAMDA(NPNP2+1)
       NR=NR-1
       GO TO 606
C
C          NOISE ON OUTPUT ONLY
605    CALL SOLVE2(Z,G,GAMMA,XLAMDA,NP1,NR,MAX)
       XMEAN=XLAMDA(NPNP2+1)
       NR=NR-1
       GO TO 606
C
C          NOISE ON INPUT ONLY
618    NPP=NPNP2+1
       NR=NR-1
       DO550I=1,NPP
550    GAMMA(I)=G(I,NPP)
       DO551J=1,NR
       JJ=NR-J+1
       DO551I=1,NPP
551    G(I,NP2+JJ)=G(I,NP1+JJ)
       DO552I=1,NPP
552    G(I,NP2)=GAMMA(I)
       DO553I=1,NPP
553    GAMMA(I)=G(NPP,I)
       DO554I=1,NR
       II=NR-I+1
       DO554J=1,NPP
554    G(NP2+II,J)=G(NP1+II,J)
       DO555I=1,NPP
555    G(NP2,I)=GAMMA(I)
       CALL SOLVE3(Z,G,GAMMA,XLAMDA,NP2,NR,MAX)
       XMEAN=XLAMDA(NP2)
       DO556I=1,NR
556    XLAMDA(NP1+I)=XLAMDA(NP2+I)
606    IF(JOPT)120,130,120
120    NPNP2=NPNP2+IREM
       IF(ILEVIN.EQ.1)GO TO 124
       DO122I=1,NR
122    XLAMDA(NPNP2-I+1)=XLAMDA(NP2+NR-I)
       DO123I=1,IREM
123    XLAMDA(NP1+I)=0.0000
       GO TO 130
```

```
124     NN3MIR=NPNP2+1-IREM
        DO125I=NN3MIR,NPNP2
125     XLAMDA(I)=0.0D00
130     CONTINUE
         FAC=1.0
        IF(INORM.EQ.1)FAC=VARQ/VARW
        DO301I=NP2,NPNP2
301     XLAMDA(I)=XLAMDA(I)*FAC
        WRITE(6,1001)
1001    FORMAT(10X,'THE SYNTHETIC COEFFICIENT VECTOR, XLAMDA, IS')
        CALL PRVEC(XLAMDA,NPNP2)
        DO 150 I=1,NPNP2
150     GAMMA(I)=XLAMDA(I)
        IF(ILEVIN.EQ.1)GO TO 165
C
C           GENERATING GAMMA FROM XLAMDA
        CALL BUILDA(S,Q,DEL,N,MAX)
        DO160I=1,NPNP2
        GAMMA(I)=0.0D00
        FAC=1.0
        IF(INORM.EQ.1)FAC=VARQ
        DO160J=1,NPNP2
160     GAMMA(I)=GAMMA(I)+S(I,J)*XLAMDA(J)
        XMEAN=XMEAN*S(1,NP1)*FAC/GAMMA(1)
165     CONTINUE
        WRITE(6,655)XMEAN
655     FORMAT(/1X,'MEAN COEFFICIENT IS ',D13.6,//)
        DO200I=2,NPNP2
200     GAMMA(I)=GAMMA(I)/GAMMA(1)
        GAMMA(1)=1.0D00
        IF(IDLY.EQ.0)GO TO 172
        IDLY1=IDLY+1
        DO 170 II=IDLY1,NP1
        I=NPNP2+1-II
170     GAMMA(I+IDLY)=GAMMA(I)
        DO 172 I=1,IDLY
        GAMMA(I+NP1)=0.0D00
172     CONTINUE
C
C           CALCULATING THE EQUIVALENT CONTINUOUS DESCRIPTION
        CALL IZTOS(GAMMA,N,DELTA,IZTS)
        WRITE(6,1003)
1003    FORMAT(///,1X,100(1H-),/,1X,100(1H-))
        RETURN
        END
```

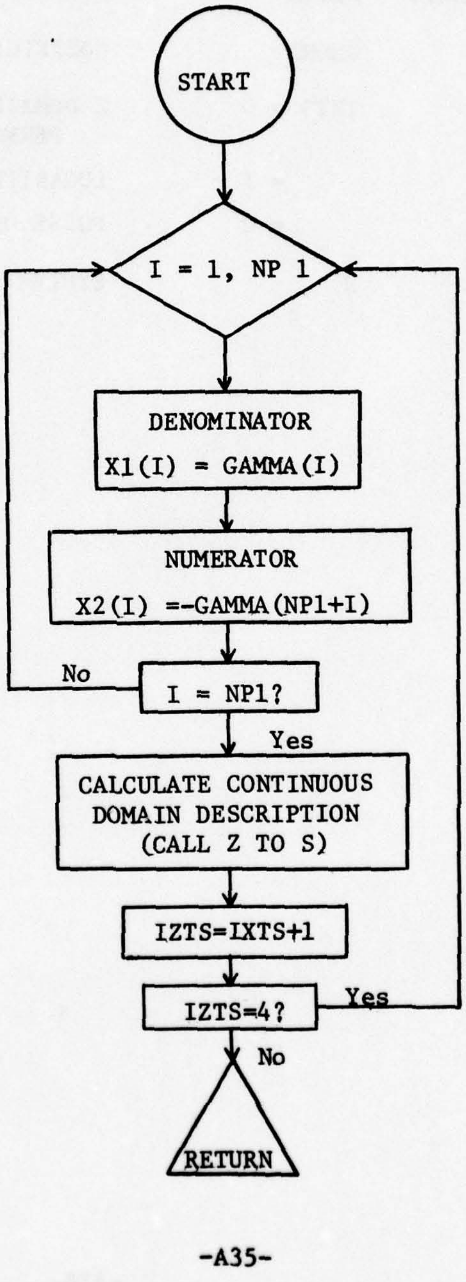
SUBROUTINE: IZTOS

PURPOSE: SEPARATES THE NUMERATOR FROM THE DENOMINATOR PARAMETERS IN GAMMA.

EQUATION: DENOMINATOR; X1(I) = GAMMA(I)

NUMERATOR; X2(I) = GAMMA(NP1 + I)

FLOW CHART:



SUBROUTINE: IZTOS

DESCRIPTION: This subroutine takes the coefficient vector GAMMA and separates the vector into the numerator (X2(I) array) and denominator (X1(I) array).

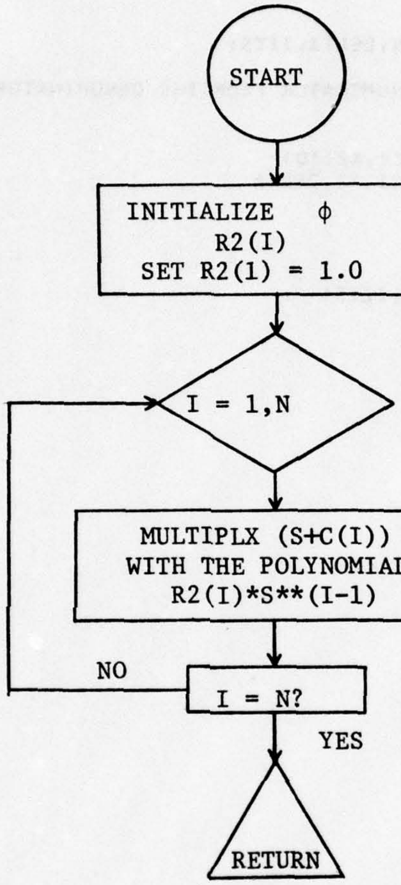
PROGRAM VARIANCE:	DELTA	SAMPLING INTERVAL
	GAMMA	COEFFICIENT VECTOR
	IZTS = 0	Z DOMAIN TO S DOMAIN CONVERSION NOT PERFORMED
	= 1	LOGARITHMIC TRANSFORMATION IS PERFORMED
	= 2	PULSE DELAYED TRANSFORMATION IS PERFORMED
	N	SYSTEM ORDER

```
      SUBROUTINE IZTOS(GAMMA,N,DELTA,IZTS)
C
C        IZTOS SEPARATES THE NUMERATOR FROM THE DENOMINATOR PARAMETERS
C        IN GAMMA
C
      DIMENSION GAMMA(1),X1(10),X2(10)
      DOUBLE PRECISION GAMMA,X1,X2,DELTA
      NP1=N+1
200   DO3I=1,NP1
      X1(I)=GAMMA(I)
3     X2(I)=-GAMMA(NP1+I)
      CALL ZTOS(X1,X2,N,DELTA,IZTS)
      IZTS=IZTS+1
      IF(IZTS.EQ.4) GO TO 200
      RETURN
      END
```

SUBROUTINE: POLCON

PURPOSE: CONSTRUCTS POLYNOMIAL FROM ITS ROOTS

FLOW CHART:



DESCRIPTION: This subroutine constructs a polynomial from its roots and the polynomial coefficients are stored in an array R2(I) in ascending order.

PROGRAM VARIABLES

C	ROOTS USED TO FORM POLYNOMIAL
K	OPTION WHICH SUPPRESSES POLYNOMIAL CONSTRUCTION WITH SPECIFIED ROOT(S)
N	ORDER OF SYSTEM
R2	COEFFICIENTS OF POLYNOMIAL

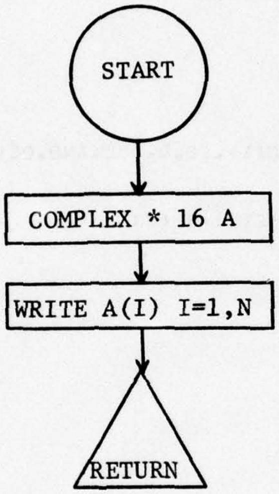
```
      SUBROUTINE POLCON(C,R2,K,N)
C
C     A POLYINOMIAL CONSTRUCTION PROGRAM NEEDED FOR ZTOS
C
      DIMENSION C(1),R2(1)
      COMPLEX*16 C,R2,COMP
      REAL*8 DC(2)
      EQUIVALENCE (COMP,DC)
      NP1=N+1
      DO10I=2,NP1
10    R2(I)=0.0D00
      R2(1)=1.0D00
      DO4I=1,N
      COMP=C(I)
      IF(I.EQ.K.OR.(DC(1).EQ.0.0D0.AND.DC(2).EQ.0.0D0))GO TO 4
      DO2JJ=1,I
      J=I-JJ+1
2     R2(J+1)=R2(J+1)*C(I)+R2(J)
      R2(1)=R2(1)*C(I)
4     CONTINUE
      RETURN
      END
```

SUBROUTINE: PRCVEC

PURPOSE: This subroutine prints out a complex single dimensioned array.

EQUATION: Complex number A + BJ is printed (A, BJ)

FLOW CHART:



DESCRIPTION: This subroutine is called in the ZTOS subroutine when the poles and zeroes in the S domain are needed.

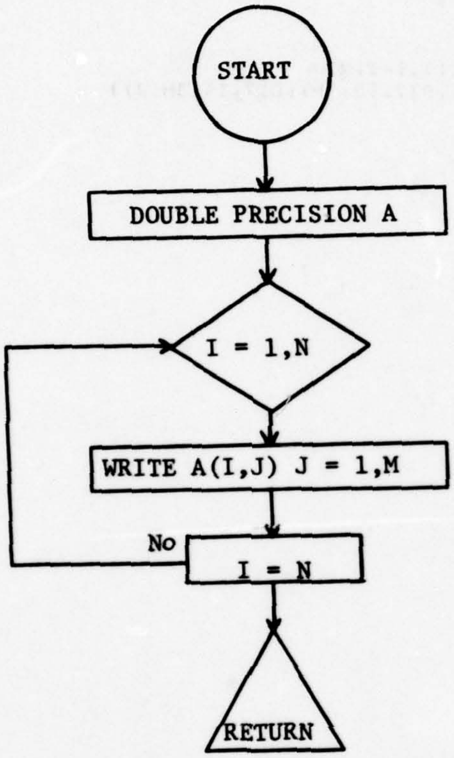
PROGRAM VARIABLES:		
	A	ARRAY TO BE OUTPUT
	N	NUMBER OF ELEMENT IN ARRAY

```
      SUBROUTINE PRCVEC(A,N)
C
C     THIS SUBROUTINE PRINTS OUT A COMPLEX SINGLE DIMENSIONED ARRAY
C        A COMPLEX NUMBER OF THE FORM A + B J IS OUTPUTTED IN THE FORM
C                  ( A, B J)      WHERE J = SQUARE ROOT OF -1
      DIMENSION A(1)
      COMPLEX*16 A
      WRITE(6,2)
      WRITE(6,1)(A(I),I=1,N)
1     FORMAT(1X,1H(,D17.10,1H,,D17.10,3H J))
      WRITE(6,2)
      WRITE(6,2)
2     FORMAT(/)
      RETURN
      END
```

SUBROUTINE: PRMAT

PURPOSE: SUBROUTINE OUTPUTS DOUBLE PRECISION DOUBLE DIMENSION ARRAY

FLOW CHART:



DESCRIPTION: This subroutine is called in GRAM II and takes an array of two dimensions and gives an output of the same two dimensional array in double precision.

PROGRAM VARIABLES:

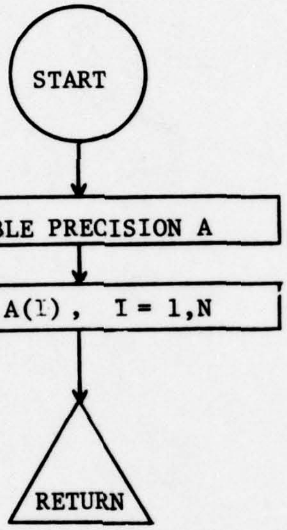
A	OUTPUT DOUBLE PRECISION ARRAY
M	MATRIX "A" COLUMN DIMENSION
N	MATRIX "A" ROW DIMENSION
NMAX	DIMENSION SIZE OF TWO DIMENSIONAL ARRAY

```
      SUBROUTINE PRMAT(A,N,M,NMAX)
      DOUBLE PRECISION A
C
C     THIS SUBROUTINE OUTPUTS DOUBLE PRECISION DOUBLE DIMENSIONED ARRAY
      DIMENSION A(NMAX,1)
      WRITE(6,1)
      DO2I=1,N
2     WRITE(6,3)(A(I,J),J=1,M)
3     FORMAT(1X,10D13.5)
      WRITE(6,1)
      WRITE(6,1)
1     FORMAT(/)
      RETURN
      END
```

SUBROUTINE: PRVEC

PURPOSE: SUBROUTINE OUTPUTS DOUBLE PRECISION SINGLE DIMENSION ARRAY

FLOW CHART:



DESCRIPTION: This subroutine is called in GRAM_{II} and other routines to print one-dimensional arrays in double precision.

PROGRAM VARIABLES:	A	ARRAY THAT IS OUTPUT IN DOUBLE PRECISION
	N	NUMBER OF ELEMENTS IN ARRAY

```
      SUBROUTINE PRVEC(A,N)
C
C     THIS SUBROUTINE OUTPUTS DOUBLE PRECISION SINGLE DIMENSIONED ARRAY
      DIMENSION A(1)
      DOUBLE PRECISION A
      WRITE(6,31)
      WRITE(6,1)(A(I),I=1,N)
1     FORMAT(1X,10D13.5)
      WRITE(6,31)
      WRITE(5,31)
31    FORMAT(/)
      RETURN
      END
```

SUBROUTINE: RESPON

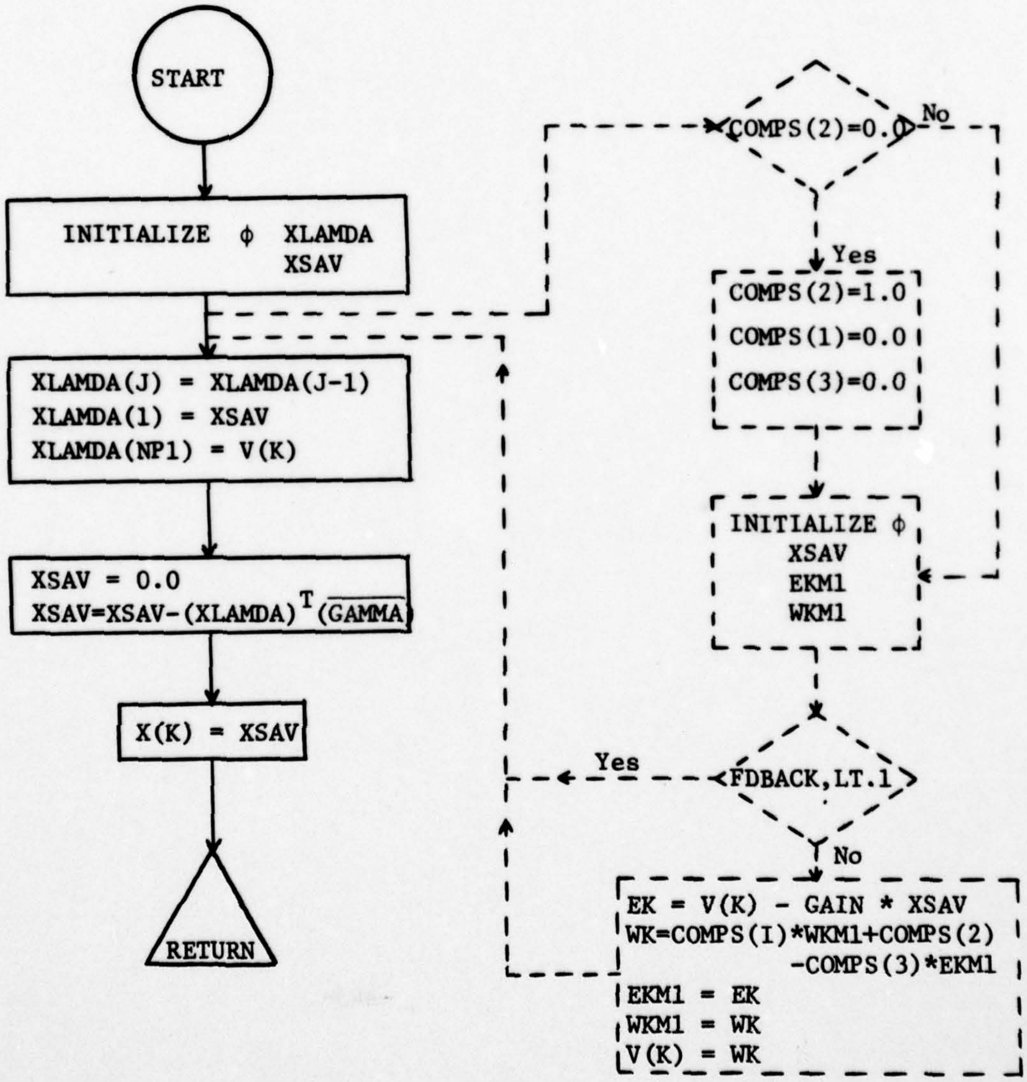
PURPOSE: CALCULATES RESPONSE (X_K) FROM COEFFICIENT VECTOR (GAMMA) TIMES THE ARRAY XLAMDA.

EQUATIONS: $[X_K,\ (\mathrm{XLAMDA})]\ [\ \mathrm{GAMMA}\ ] = 0$

$$\text{or } X_K = -\left[X_{k-1} \ldots X_{k-n}\quad V_K \cdot\ \cdot\ V_{k-n}\right] \begin{bmatrix}\gamma(2)\\ \cdot\\ \cdot\\ \cdot\\ \gamma(N+N+2)\end{bmatrix}$$

$$\text{also } X_K = -\ (\mathrm{XLAMDA})^T\ (\overline{\mathrm{GAMMA}})$$

FLOW CHART:



SUBROUTINE: RESPON

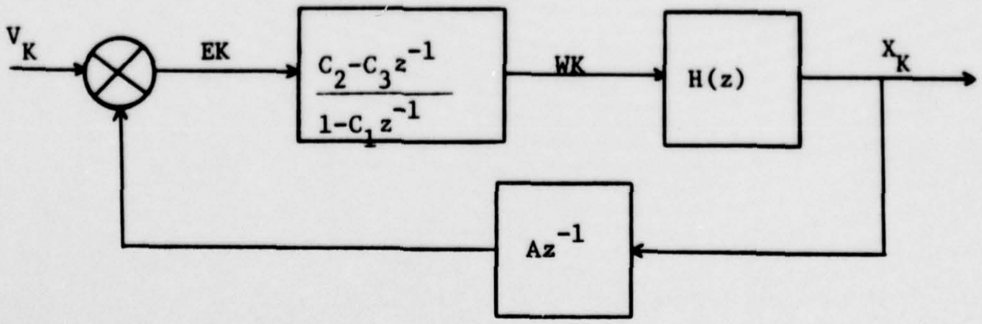
DESCRIPTION: Subroutine RESPON determines the response of

$$H(z) = \frac{b_o + b_1z^{-1} + b_2z^{-2} + ... + b_nz^{-n}}{1 + a_1z^{-1} + a_2z^{-2} + ... + a_nz^{-n}}$$ to the

input sequence V(K). The coeffieints are entered as an NPNP2 = N + N + 2 vector GAMMA = $(1, a_1, ..., a_n, - b_0, ..., -b_n)$.

PROGRAM VARIABLES:		
	FDBACK	NO FEEDBACK FDBACK = 0 NEGATIVE FEEDBACK FDBACK = 1
	GAMMA	COEFFICIENT VECTOR
	MP1	NUMBER OF DATA POINTS
	N	MODEL ORDER
	V	INPUT SEQUENCE
	X	OUTPUT SEQUENCE
	XLAMDA	WORKING ARRAY

FEEDBACK AND COMPENSATION:



SUBROUTINE: RESPON

PURPOSE: CALCULATES RESPONSE (X_K) FOR THE SYSTEM SHOWN IN ABOVE FIGURE. THIS ADDITION TO SUBROUTINE RESPON INCORPORATES THE FLEXABILITY OF ADDING NEGATIVE FEEDBACK AND CASCADE COMPENSATOR IN THE FORWARD LOOP FOR OPTIMIZATION.

EQUATION: $EK = V_K - AX_{k-1}$

WK = WKM1 * COMPS (1) + EK * COMPS (2) - COMPS (3)*EKM1

FLOW CHART: (See flow chart for RESPON the dotted section is for Feedback and Compensation network.)

BEST AVAILABLE COPY

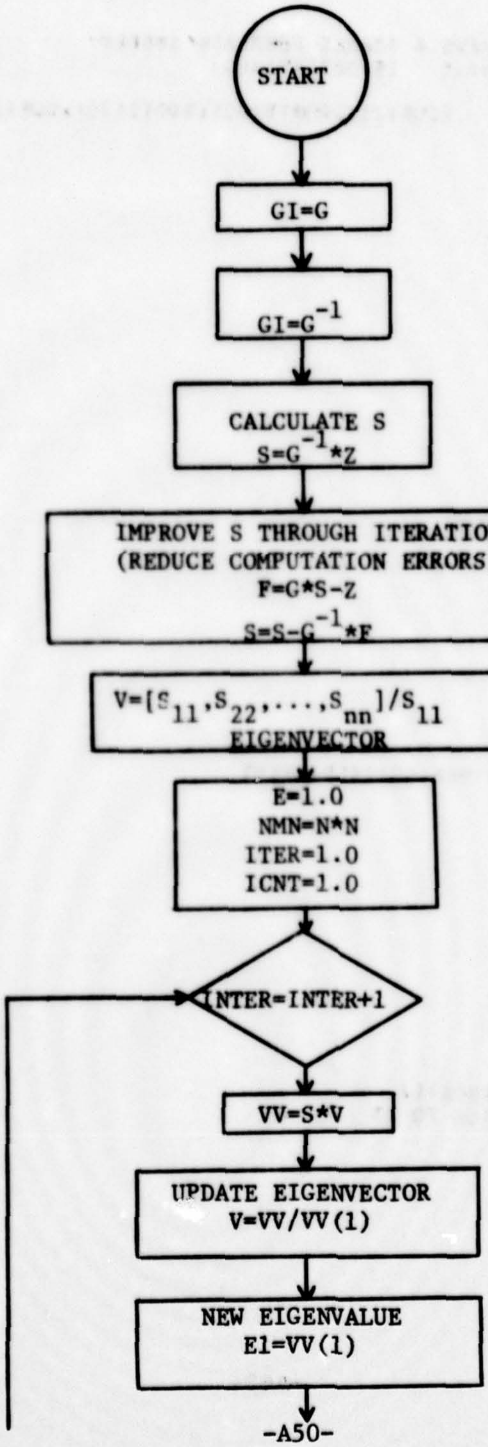
```
      SUBROUTINE RESPON(X,V,N,GAMMA,XLAMDA,MP1,FDBACK)
      DIMENSION X(1),V(1),GAMMA(1),XLAMDA(1)
      REAL*8 XSAV,GAMMA,XLAMDA,GAIN
      INTEGER FDBACK

       THESE CARDS FOR DETERMINING A STABLE FEEDBACK SYSTEM
      FEEDBACK GAIN CALLED  'GAIN'   IS DETERMINED

      REAL*8 CN,GMN,GMX,TEM,      XCOF(21),ROOTR(20),ROOTI(20),COF(21)
      REAL*8 COMPS
      COMMON /COMPEN/COMPS(10)
      GAIN=-.18
      NP1=N+1
      NP2=N+2
      ISTABL=0




      NM1=N-1
      NP1=N+1
      NPNP1=N+N+1
      NPNP2=N+N+2
      DO19I=1,NPNP1
19    XLAMDA(I)=0.0D0
      IF(COMPS(2).NE.0.)GO TO 899
      COMPS(2)=1.0
      COMPS(1)=0.0
      COMPS(3)=0.0
899   CONTINUE
      XSAV=0.0D00
      EKM1=0.
      WKM1=0.
      DO20K=1,MP1
      IF(FDBACK.LT.1)GO TO 211
      EK=V(K)-GAIN*XSAV
      WK=COMPS(1)*WKM1+COMPS(2)*EK-COMPS(3)*EKM1
      WKM1=WK
      EKM1=EK
      V(K)=WK
211   DO21I=1,NM1
      J=NP1-I
21    XLAMDA(J)=XLAMDA(J-1)
      DO22I=1,N
      J=NPNP2-I
22    XLAMDA(J)=XLAMDA(J-1)
      XLAMDA(1)=XSAV
      XLAMDA(NP1)=V(K)
      XSAV=0.0D00
      DO23I=1,NPNP1
23    XSAV=XSAV-GAMMA(I+1)*XLAMDA(I)
      IF(DABS(XSAV).GT.1.0D10)GO TO 27
20    X(K)=XSAV
      RETURN



27    DO28I=K,MP1
28    X(I)=0.0
      RETURN
      END
```

SUBROUTINE: SOLVE1

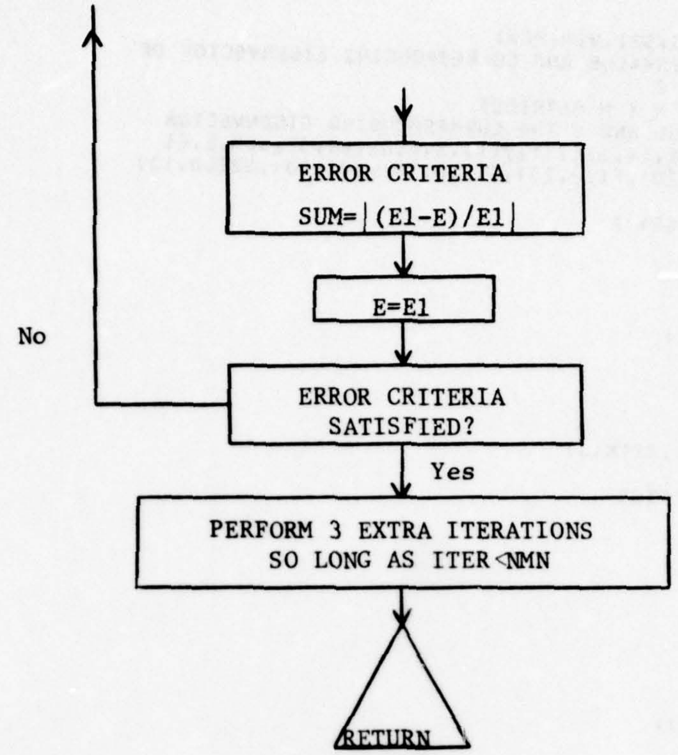
PURPOSE: FINDS MAXIMUM EIGENVALUE AND CORRESPONDING EIGENVECTOR OF

(Z-E*G)*V=0

FLOW CHART:



SUBROUTINE: SOLVE1


ERROR CRITERIA
SUM=|(E1-E)/E1|
E=E1
No
ERROR CRITERIA SATISFIED?
Yes
PERFORM 3 EXTRA ITERATIONS SO LONG AS ITER<NMN
RETURN

DESCRIPTION: Subroutine SOLVE1 uses the first quadrant of the GRAM matrix and noise correction matrix " " to calculate the output portion of the EIGENVECTOR (N+1 elements) plus the maximum EIGENVALUE.

PROGRAM VARIABLES:

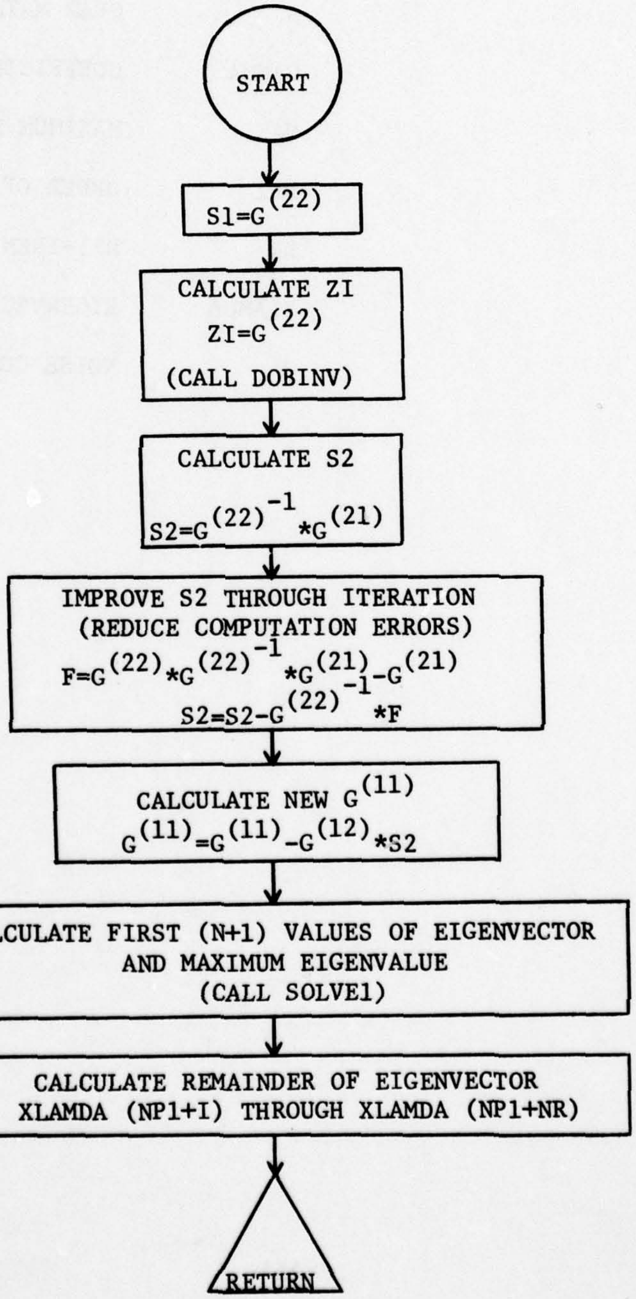
G	FIRST QUADRANT OF GRAM MATRIX
MAX	MAXIMUM ROWS PERMISSIBLE
N	N+1 (ORDER OF SYSTEM + 1)
S21	COEFFICIENT MATRIX
V	EIGENVECTOR
Z	NOISE CORRECTION MATRIX

```
      SUBROUTINE SOLVE1(Z,G,S21,V,N,MAX)
C        FINDS MAXIMUM EIGENVALUE AND CORRESPONDING EIGENVECTOR OF
C        ( Z - E*G ) * V = 0
C        WHERE Z AND G ARE N X N MATRICES
C        E IS THE EIGENVALUE AND V THE CORRESPONDING EIGENVECTOR
      REAL*8 Z(MAX,1),G(MAX,1),S21(1),V(1),S,F,GI,VV,S2,SUM,E,E1
      COMMON /MATRIX/S(20,20),F(20,20),GI(20,20),VV(100),S2(10,10)
C
C     CALCULATE S=(G INVERSE)*Z
C
      DO1I=1,N
      DO1J=1,N
1     GI(I,J)=G(I,J)
      CALL DOBINV(GI,N,MAX)
      DO2I=1,N
      DO2J=1,N
      S(I,J)=0.0D0
      DO2K=1,N
2     S(I,J)=S(I,J)+GI(I,K)*Z(K,J)
C
C     IMPROVE S THRU ITERATION
C     F=G*S-Z
C     S=S-(G INVERSE)*F
C
      ND2=N
      DO100ITER=1,ND2
      DO3I=1,N
      DO3J=1,N
      SUM=0.0D0
      DO4K=1,N
4     SUM=SUM+G(I,K)*S(K,J)
3     F(I,J)=SUM-Z(I,J)
      DO5I=1,N
      DO5J=1,N
      SUM=0.0D0
      DO6K=1,N
6     SUM=SJM+GI(I,K)*F(K,J)
5     S(I,J)=S(I,J)-SUM
100   CONTINUE
C
C        INITIALLY V=(S(1,1), . . ., S(N,N))/S(1,1), E=1
C       ITERATE: EVEC  VV=S*V, EVAL E1=VV(1)
C
      DO7I=1,N
7     V(I)=S(I,I)/S(1,1)
      E=1.0D0
      NMN=N*N
      ITER=1
      ICNT=1
8     ITER=ITER+1
      DO9I=1,N
      VV(I)=0.0D0
      DO9J=1,N
9     VV(I)=VV(I)+S(I,J)*V(J)
      DO10I=1,N


10    V(I)=VV(I)/VV(1)
      E1=VV(1)
      SUM=DABS((E1-E)/E1)
      E=E1
      IF(SUM.GT.1.0D-8.AND.ITER.LT.NMN)GO TO 8
      ICNT=ICNT+1
      IF(ICNT.LT.5.AND.ITER.LT.NMN)GO TO 8
      WRITE(6,11)ITER,E,SUM
11    FORMAT(10X,'ITER=',I3,'  MAX.EIGENVALUE=',D13.6,'  ERROR=',D13.6)
      RETURN
      END
```

SUBROUTINE: SOLVE2

PURPOSE: CALCULATE MAXIMUM EIGNEVALUE AND EIGENVECTOR

FLOW CHART:


START
$S1=G^{(22)}$
CALCULATE ZI
$ZI=G^{(22)}$
(CALL DOBINV)
CALCULATE S2
$S2=G^{(22)^{-1}}*G^{(21)}$
IMPROVE S2 THROUGH ITERATION
(REDUCE COMPUTATION ERRORS)
$F=G^{(22)}*G^{(22)^{-1}}*G^{(21)}-G^{(21)}$
$S2=S2-G^{(22)^{-1}}*F$
CALCULATE NEW $G^{(11)}$
$G^{(11)}=G^{(11)}-G^{(12)}*S2$
CALCULATE FIRST (N+1) VALUES OF EIGENVECTOR
AND MAXIMUM EIGENVALUE
(CALL SOLVE1)
CALCULATE REMAINDER OF EIGENVECTOR
XLAMDA (NP1+I) THROUGH XLAMDA (NP1+NR)
RETURN

SUBROUTINE: SOLVE2

DESCRIPTION: Subroutine SOLVE2 calculates the maximum EIGENVALUE and EIGENVECTOR when noise is present on both input and output or the noise is present only on the output. SOLVE2 uses subroutine SOLVE1 to calculate the first (N+1) EIGENVALUES and the maximum EIGENVALUE. SOLVE2 uses an interation process (similar to SOLVE1 and SOLVE3) to reduce computation errors.

PROGRAM VARIABLES:

G	GRAM MATRIX
GAMMA	COEFFICIENT MATRIX
MAX	MAXIMUM ROWS PERMISSIBLE
NP1	ORDER OF SYSTEM+1 (N+1)
NR	NP1-IREM
XLAMDA	EIGENVECTOR
Z	NOISE CORRECTION MATRIX

AD-A050 206 UNIVERSITY OF SOUTH FLORIDA TAMPA F/G 9/2

THEORY AND COMPUTER PROGRAM FOR TRANSFER FUNCTION IDENTIFICATIO--ETC(U)

FEB 78 V K JAIN, G J DOBECK, L J LAWDERMILT N61331-75-C-0012

UNCLASSIFIED NCSL-TM-204-78 NL

2 OF 2

AD
A050 206

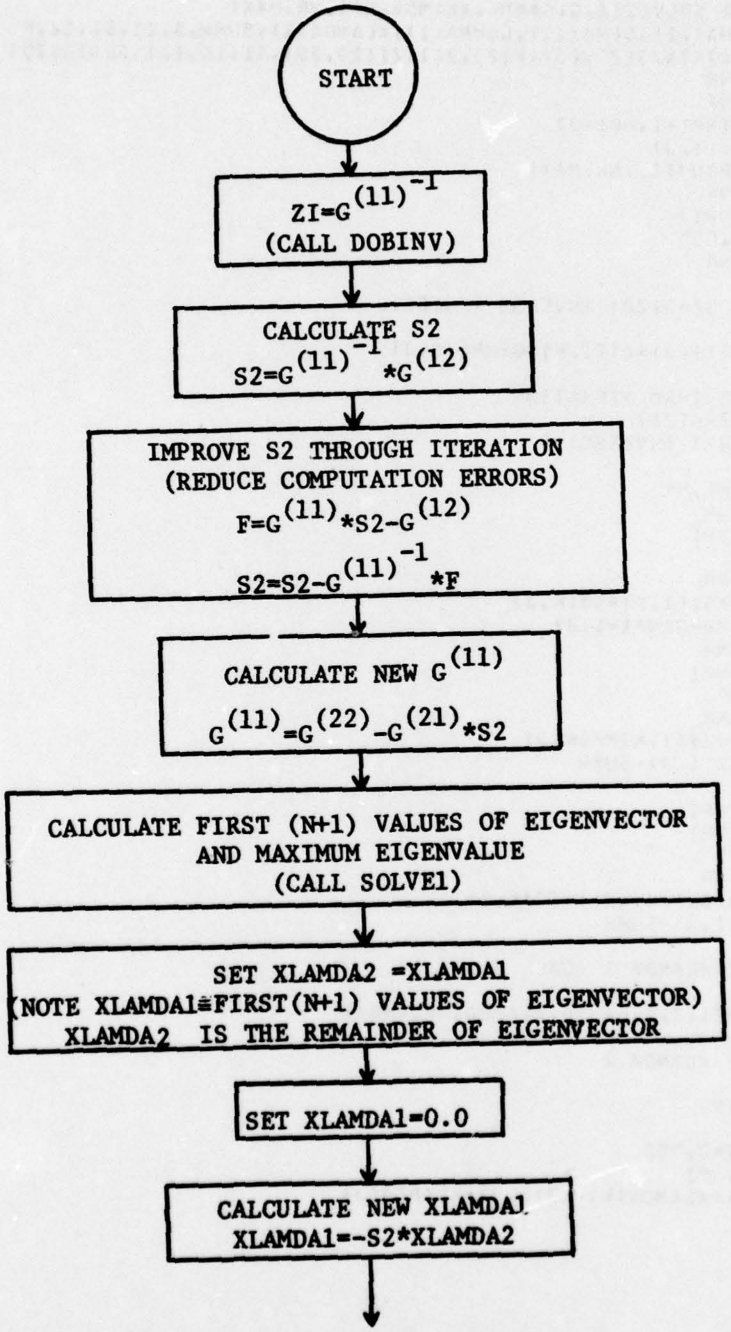
END
DATE
FILMED
3-78
DDC

```
      SUBROUTINE SOLVE2(Z,G,GAMMA,XLAMDA,NP1,NR,MAX)
      REAL*8 Z(MAX,1),G(MAX,1),GAMMA(1),XLAMDA(1),SUMW,S,ZI,S1,S2,F
      COMMON /MATRIX/S(20,20),F(20,20),ZI(20,20),S1(10,10),S2(10,10)
      DO620I=1,NR
      DO620J=1,NR
      ZI(I,J)=G(NP1+I,NP1+J)
620   S1(I,J)=ZI(I,J)
        CALL DOBINV(ZI,NR,MAX)
      DO621I=1,NR
      DO621J=1,NP1
      S2(I,J)=0.0D0
      DO621K=1,NR
C
C     CALCULATE S2=G(22) INVERSE * G(21)
C
621   S2(I,J)=S2(I,J)+ZI(I,K)*G(NP1+K,J)
C
C     IMPROVE S2 THRU ITERATION
C     F=G(22)*S2-G(21)
C     S2=S2-(G(22) INVERSE)*F
C
      DO624ITER=1,NR
      DO622I=1,NR
      DO622J=1,NP1
      SUMW=0.0D0
      DO623K=1,NR
623   SUMW=SUMW+S1(I,K)*S2(K,J)
622   F(I,J)=SUMW-G(NP1+I,J)
      DO625I=1,NR
      DO625J=1,NP1
      SUMW=0.0D0
      DO626K=1,NR
626   SUMW=SUMW+ZI(I,K)*F(K,J)
625   S2(I,J)=S2(I,J)-SUMW
624   CONTINUE
      DO627I=1,NP1
      DO627J=1,NP1
      SUMW=0.0D0
      DO628K=1,NR
628   SUMW=SUMW+G(I,K+NP1)*S2(K,J)
627   G(I,J)=G(I,J)-SUMW
C
C     CALCULATE XLAMDA 1 (CALL SOLVE 1)
C
      CALL SOLVE1(Z,G,GAMMA,XLAMDA,NP1,MAX)
C
C     CALCULATE XLAMDA 2
C
      DO629I=1,NR
      K=NP1+I
      XLAMDA(K)=0.0D0
      DO629J=1,NP1
629   XLAMDA(K)=XLAMDA(K)-S2(I,J)*XLAMDA(J)
      RETURN
      END
```

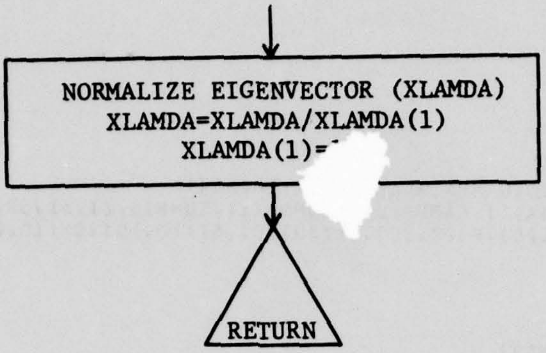
SUBROUTINE: SOLVE 3

PURPOSE: CALCULATE MAXIMUM EIGENVALUE AND CORRESPONDING EIGENVECTOR (NOISE ON INPUT ONLY)

FLOW CHART:


START
ZI=$G^{(11)^{-1}}$
(CALL DOBINV)
CALCULATE S2
S2=$G^{(11)^{-1}}$*$G^{(12)}$
IMPROVE S2 THROUGH ITERATION
(REDUCE COMPUTATION ERRORS)
F=$G^{(11)}$*S2-$G^{(12)}$
S2=S2-$G^{(11)^{-1}}$*F
CALCULATE NEW $G^{(11)}$
$G^{(11)}$=$G^{(22)}$-$G^{(21)}$*S2
CALCULATE FIRST (N+1) VALUES OF EIGENVECTOR
AND MAXIMUM EIGENVALUE
(CALL SOLVE1)
SET XLAMDA2 =XLAMDA1
(NOTE XLAMDA1≡FIRST(N+1) VALUES OF EIGENVECTOR)
XLAMDA2 IS THE REMAINDER OF EIGENVECTOR
SET XLAMDA1=0.0
CALCULATE NEW XLAMDA1
XLAMDA1=-S2*XLAMDA2

SUBROUTINE: SOLVE3



DESCRIPTION: SOLVE3 is used to calculate the EIGENVECTOR when the system has noise only on input. SOLVE3 uses SOLVE1 to calculate the maximum EIGENVALUE and the (N+1) EIGENVALUES . The remaining EIGENVECTOR elements are calculated in subroutine SOLVE3. SOLVE3 uses an iteration process (similar to SOLVE1 an SOLVE2) to reduce computation errors.

PROGRAM VARIABLES:

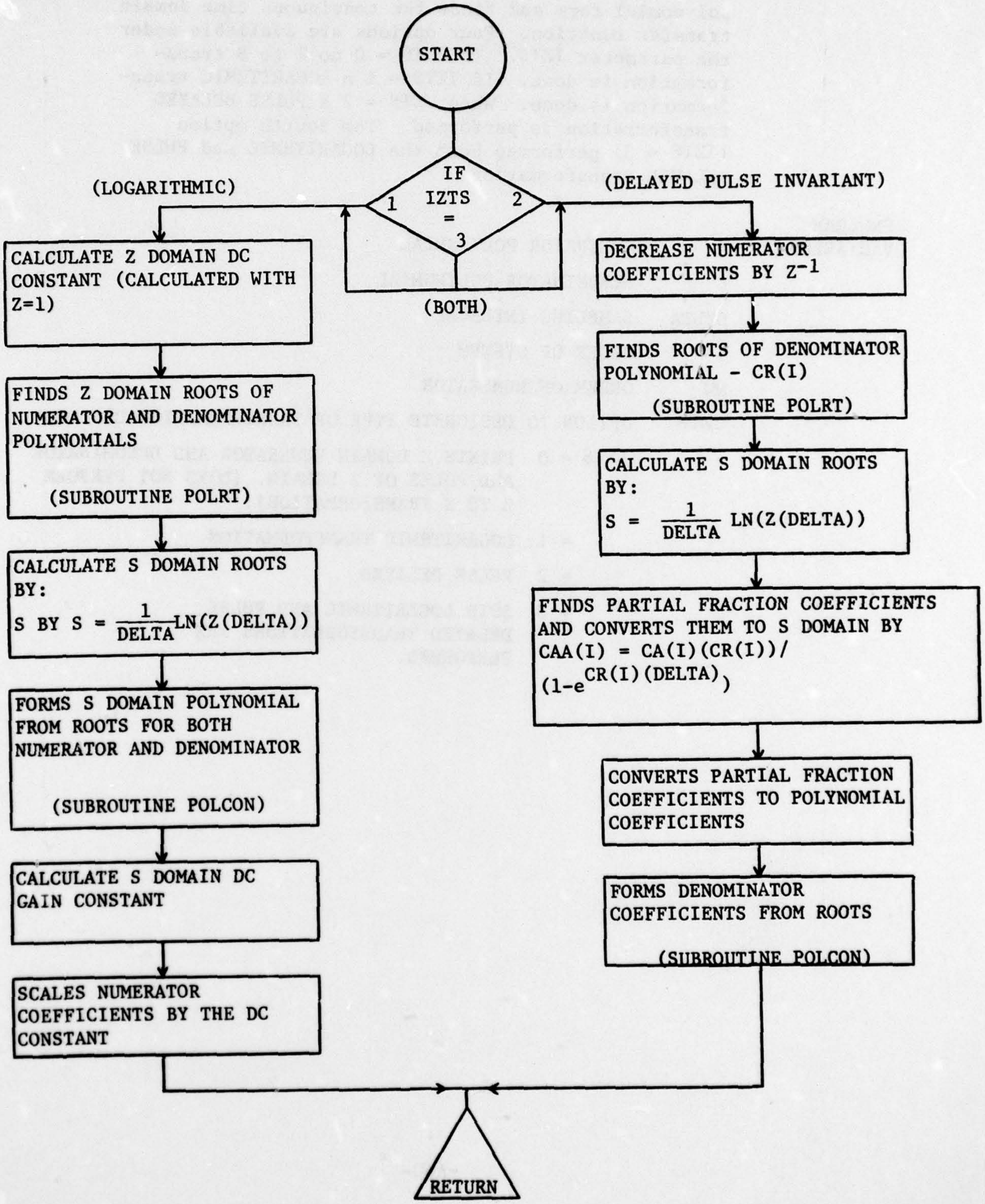
G	GRAM MATRIX
GAMMA	COEFFICIENT MATRIX
MAX	MAXIMUM ROWS PERMISSIBLE
NP1	SYSTEM ORDER +1,(N+1)
NR	NP1-IREM
XLAMDA	EIGENVECTOR
Z	NOISE CORRECTION MATRIX

```
      SUBROUTINE SOLVE3(Z,G,GAMMA,XLAMDA,NP1,NR,MAX)
      REAL*8 Z(MAX,1),G(MAX,1),GAMMA(1),XLAMDA(1),SUMW,S,ZI,S1,S2,F
      COMMON /MATRIX/S(20,20),F(20,20),ZI(20,20),S1(10,10),S2(10,10)
      NPNP2=NP1+NR
      DO1I=1,NP1
      DO1J=1,NP1
1     ZI(I,J)=G(I,J)
      CALL DOBINV(ZI,NP1,MAX)
      DO2I=1,NP1
      DO2J=1,NR
      S2(I,J)=0.0D0
      DO2K=1,NP1
2     S2(I,J)=S2(I,J)+ZI(I,K)*G(K,J+NP1)
      DO3ITER=1,NP1
      DO4I=1,NP1
      DO4J=1,NR
      SUMW=0.0D0
      DO5K=1,NP1
5     SUMW=SUMW+G(I,K)*S2(K,J)
4     F(I,J)=SUMW-G(I,J+NP1)
      DO6I=1,NP1
      DO6J=1,NR
      SUMW=0.0D0
      DO7K=1,NP1
7     SUMW=SUMW+ZI(I,K)*F(K,J)
6     S2(I,J)=S2(I,J)-SUMW
3     CONTINUE
      DO8I=1,NR
      DO8J=1,NR
      SUMW=0.0D0
      DO9K=1,NP1
9     SUMW=SUMW+G(I+NP1,K)*S2(K,J)
8     G(I,J)=G(I+NP1,J+NP1)-SUMW
      CALL SOLVE1(Z,G,GAMMA,XLAMDA,NR,MAX)
      DO10I=1,NR
10    XLAMDA(NP1+I)=XLAMDA(I)
      DO11I=1,NP1
      XLAMDA(I)=0.0D0
      DO11J=1,NR
11    XLAMDA(I)=XLAMDA(I)-S2(I,J)*XLAMDA(J+NP1)
      DO12I=2,NPNP2
12    XLAMDA(I)=XLAMDA(I)/XLAMDA(1)
      XLAMDA(1)=1.0D0
      RETURN
      END
```

SUBROUTINE: ZTOS

PURPOSE: CONVERTS A DISCRETE TIME TRANSFER FUNCTION H(Z) TO A CONTINUOUS TIME TRANSFER FUNCTION H(S).

FLOW CHART:


START
IF IZTS =
1
2
3
(LOGARITHMIC)
(DELAYED PULSE INVARIANT)
(BOTH)
CALCULATE Z DOMAIN DC CONSTANT (CALCULATED WITH Z=1)
FINDS Z DOMAIN ROOTS OF NUMERATOR AND DENOMINATOR POLYNOMIALS (SUBROUTINE POLRT)
CALCULATE S DOMAIN ROOTS BY: S BY S = 1/DELTA LN(Z(DELTA))
FORMS S DOMAIN POLYNOMIAL FROM ROOTS FOR BOTH NUMERATOR AND DENOMINATOR (SUBROUTINE POLCON)
CALCULATE S DOMAIN DC GAIN CONSTANT
SCALES NUMERATOR COEFFICIENTS BY THE DC CONSTANT
DECREASE NUMERATOR COEFFICIENTS BY Z^{-1}
FINDS ROOTS OF DENOMINATOR POLYNOMIAL - CR(I) (SUBROUTINE POLRT)
CALCULATE S DOMAIN ROOTS BY: S = 1/DELTA LN(Z(DELTA))
FINDS PARTIAL FRACTION COEFFICIENTS AND CONVERTS THEM TO S DOMAIN BY CAA(I) = CA(I)(CR(I))/ $(1-e^{CR(I)(DELTA)})$
CONVERTS PARTIAL FRACTION COEFFICIENTS TO POLYNOMIAL COEFFICIENTS
FORMS DENOMINATOR COEFFICIENTS FROM ROOTS (SUBROUTINE POLCON)
RETURN

SUBROUTINE ZTOS

DESCRIPTION: This subroutine uses the H(Z) transfer function in polynomial form and finds the continuous time domain transfer function. Four options are available under the parameter IZTS. If IZTS = 0 no Z to S transformation is done. If IZTS = 1 a LOGARITHMIC transformation is done. When IZTS = 2 a PULSE DELAYED transformation is performed. The fourth option (IZTS = 3) performes both the LOGARITHMIC and PULSE DELAYED transformations.

PROGRAM VARIABLES:

A	NUMERATOR POLYNOMIAL	
B	DENOMINATOR POLYNOMIAL	
DELTA	SAMPLING INTERVAL	
N	ORDER OF SYSTEM	
NN	ORDER OF NUMERATOR	
IZTS	OPTION TO DESIGNATE TYPE OF TRANSFORMATION DESIRED	
	IZTS = 0	PRINTS Z DOMAIN NUMERATOR AND DENOMINATOR AND POLES OF Z DOMAIN. (DOES NOT PERFORM Z TO S TRANSFORMATION).
	= 1	LOGARITHMIC TRANSFORMATION
	= 2	PULSE DELAYED
	= 3	BOTH LOGARITHMIC AND PULSE DELAYED TRANSFORMATIONS ARE PERFORMED.

```
      SUBROUTINE ZTOS(B,A,N,DELTA,IZTS)
      COMMON NN
C
C
C     CONVERSION OF A DISCRETE TIME SYSTEM H(Z) TO A CONTINUOUS TIME SYSTEM H(S)
C
C
C     H(Z)=(A(1) + A(2)*ZETA +....)/(1 + B(2)*ZETA +....)
C                    ZETA = 1/Z
C
C     H(S)=(A(1) + A(2)*S + ....... + A(N+1)*S**N)/DENOM
C
C               DENOM=B(1) + B(2)*S + ..... + B(N+1)*S**N
C
C                    B(1) = 1 ALWAYS
C
C
      DIMENSION B(9),A(9),TEMP(20),RR(20),RI(20),CR(20),CA(20),CAA(20),
     1CA1(20),CB(2^),CF(20),CF1(20),CG(20)
      COMPLEX*16 CA,CAA,CA1,CB,CR,CON1,CON2,CONT,FAC,A1,A2,B1,B2,AA1,BB1
     1CG,CF1,CF
      REAL*8 B,A,TEMP,RR,RI,DELTA
      CONT=0.0D00
      IORP=IZTS
      NP1=N+1
      NNP1=NN+1
      A1=0.0D0
      B1=0.0D0
      DO 30I=1,NP1
      IF(I.LE.NNP1)A1=A1+A(I)
30    B1=B1+B(I)
            WRITE(6,989) NN,NNP1
989   FORMAT(10X,'NN=',I5,5X,'NNP1=',I5)
999   FORMAT(///)
      WRITE(6,999)
      WRITE(6,1000)
1000  FORMAT(' Z-DOMAIN DENOMINATOR')
      CALL PRVEC(B,NP1)
      WRITE(6,1001)
1001  FORMAT(' Z-DOMAIN NUMERATOR')
      CALL PRVEC(A,NP1)
      IF(IZTS.EQ.0) GO TO 909
      IF(IZTS.EQ.1) GO TO 200
      IF(IZTS.EQ.2) GO TO 250
      IF(IZTS.EQ.3) GO TO 200
      IF(IZTS.EQ.4) GO TO 250
200   CONTINUE
C
C
C     LOGARITHMIC TRANSFORMATION
C
C
C     WORK ON NUMERATOR
C
C
```

```
      IF(NN.EQ.0)GO TO 469
      CALL POLRT(A,TEMP,NN,RR,RI,IER)
      DO15 I=1,NN
15    CA(I)=DCMPLX(RR(I),RI(I))
      DO 7 I=1,NN
7     CA(I)=(+1.0/DELTA)*CDLOG(CA(I))
      IF(NN.EQ.N) GOTO471
469   CONTINUE
      DO 470 I=NNP1,NP1
      CAA(I)=0.0D0
470   CA(I)=0.0D0
471   CONTINUE
      IF(NN.EQ.0)CAA(1)=1.0D0
C
C
C     NOW THE FIRST NN ENTRIES OF CA CONTAIN THE S-DOMAIN ZEROES OF NUMERATOR
C     AND THE REAMINING ENTRIES ARE ZEROED OUT.
C
      IF(NN.NE.0)CALL POLCON(CA,CAA,0,N)
C
C
C     WORK ON DENOMINATOR
C
C
      CALL POLRT(B,TEMP,N,RR,RI,IER)
      DO16 I=1,N
      CR(I)=DCMPLX(RR(I),RI(I))
16    CF(I)=1.0D00/CR(I)
909   WRITE(6,1002)
      CALL PRCVEC(CF,N)
      IF(IZTS.EQ.0) GO TO 900
235   DO6 I=1,N
6     CR(I)=(-1.0/DELTA)*CDLOG(CR(I))
      WRITE(6,240)
240   FORMAT(' LOGARITHMIC TRANSFORMATION')
      WRITE(6,999)
      WRITE(6,2000)
2000  FORMAT(' POLES IN S DOMAIN')
      CALL PRCVEC(CR,N)
      DO3000I=1,N
3000  CR(I)=-CR(I)
      CALL POLCON(CR,CB,0,N)
C
C
C     ADJUST DC GAIN CONSTANT
C
C
      A2=CAA(1)
      B2=CB(1)
      FAC=(A1/B1)*(B2/A2)
      DO 603 I=1,NNP1
603   CAA(I)=CAA(I)*FAC
      GO TO 2010
C
C
C     DELAYED PULSE INVARIANT TRANSFORMATION
C
C
C     SHIFTS NUMERATOR COEFFICIENTS FOR DELAY
C
C
```

```
250   CONT = A(1)
      DO 300 I=1,N
300   A(I)=A(I+1) -CONT*B(I+1)
      A(NP1)=0.0
400   CALL POLRT(B,TEMP,N,RR,RI,IER)
      DO61I=1,N
      CR(I)=DCMPLX(RR(I),RI(I))
61    CF(I)=1.0D00/CR(I)
      WRITE(6,1002)
1002  FORMAT(1X,'THE POLES OF THE Z-DOMAIN')
      CALL PRCVEC(CF,N)
C
C
C     PARTIAL FRACTION EXPANSION
C
C
      DO3I=1,N
      CON1=1.0D00
      CON2=0.0D00
      DO4J=1,N
      CON2=CON2*CR(I)+A(N-J+1)
      IF(I-J)5,4,5
5     CON1=CON1*(1.0D00-CR(I)*CF(J))
4     CONTINUE
3     CA(I)=CON2/CON1
C
C
C     TRANSFORMATION OF DENOMINATOR AND NUMERATOR
C
C
224   DO2I=1,N
      CON1=CDLOG(CR(I))/DELTA
      CA(I)=CA(I)*CR(I)*CON1/(CR(I)-1.0D00)
2     CR(I)=CON1
      WRITE(6,241)
241   FORMAT(' DELAYED PULSE TRANSFORMATION')
      WRITE(6,999)
226   WRITE(6,1004)
1004  FORMAT(' NEGATIVE OF THE POLES IN THE S-DOMAIN')
      CALL PRCVEC(CR,N)
      WRITE(6,1003)
1003  FORMAT(1X,'NUMERATOR CONSTANTS OF FACTORIZED H(S)')
      CALL PRCVEC(CA,N)
      CALL POLCON(CR,CB,0,N)
      DO71I=1,NP1
71    CAA(I)=0.0D00
      DO9K=1,N
      CALL POLCON(CR,CF1,K,N)
      DO9J=1,N
9     CAA(J)=CAA(J)+CF1(J)*CA(K)
      CAA(NP1)=0.0D00
2010  CONTINUE
      DO450I = 1,NP1
      CAA(I)= CAA(I)+CONT*CB(I)
C
403   WRITE(6,1005)
1005  FORMAT(' S-DOMAIN DENOMINATOR')
      CALL PRCVEC(CB,NP1)
      WRITE(6,1006)
1006  FORMAT(' S-DOMAIN NUMERATOR')
      CALL PRCVEC(CAA,NP1)
      DO20I=1,NP1



      B(I)=CB(I)
20    A(I)=CAA(I)
900   RETURN
      END
```

(Reverse Page A64 Blank)

APPENDIX B

z TO s EQUIVALENCE TRANSFORMS

Logarithmic Equivalence

Equation (3) on page 4 describes a simple way of finding an s-domain transfer function H(s) corresponding to a z-domain transfer function H(z). It is implemented in the computer program whenever IZTS = 1. The basis for this correspondence lies in the logarithmic mapping $s = -\frac{1}{T}\ell n(z)$ of the poles and zeros from the z-plane to the s-plane. Appropriately, it is called the Logarithmic Equivalence Transform. The inverse mapping is similarly defined and we note that the left-hand-side of the s-plane maps into the interior of the unit circle in the z-plane.

The primary advantage of the logarithmic equivalence transform is that it preserves the degree of the numerator from the z-domain to the s-domain. However, it does not yield a good degree of invariance of the output between the continuous-time and discrete-time equivalent systems [8]. As a consequence, this method requires a finer sampling interval compared to the 'pulse interpolation' method (described below) in order to achieve a satisfactory invariance of the output.

Leading-Edge-Pulse Equivalence

This method aims for invariance of the output at the sampling instants. Strictly speaking this objective cannot be achieved for every arbitrary input because the sampled input signal loses some of the information of the orginal signal. Suitable restrictions must therefore be placed on the class of inputs for which the output invariance is sought. For example it is assumed that the bandwidth of the input signal and the highest frequency of the passband of the system are small compared to the sampling frequency (say one-tenth or smaller). Under such an assumption the input may be approximated by a train of rectangular pulses:

$$u(t) \cong \bar{u}(t) \triangleq \sum_{k=-\infty}^{\infty} u(k\Delta)p(t-k\Delta)$$

where

$$p(t) = \begin{matrix} 1 & \\ & \text{for } 0 \leq t < \Delta \\ 0 & \end{matrix}$$

Invariance of the outputs of a) H(z) excited by u(kΔ) and that of b) H(s) excited by $\bar{u}(t)$ can then be achieved by equivalencing H(z) and H(s) in the following manner:

$$H(z) = \sum_{i=1}^{n} \frac{\gamma_i}{1-\alpha_i z^{-1}} \iff \sum_{i=1}^{n} \frac{r_i}{s + p_i} = H(s),$$

$$p_i = -\frac{1}{T}\ell n(\alpha_i)$$

$$r_i = \frac{\gamma_i p_i}{(1-\alpha_i)}$$

This method yields a high degree of invariance between the outputs of a) $H(z)$ excited by $u(k\Delta)$ and b) $H(s)$ excited by the actual $u(t)$. In this respect its superiority over the logarithmic equivalence method has been demonstrated by case studies on several Navy vehicles. However, it suffers from the disadvantage that the degrees of the numerators in the z-domain and the s-domain do not, in general, equal each other.

APPENDIX C

SOLUTION OF A KEY EQUATION

As stated in Section II three different cases for equation (15) arise depending upon whether σ_q, σ_w, or both are nonzero. The solution for these different cases is discussed below.

Case 1: Noise on Both Input and Output (NSPQ ≠ 0, NSPW ≠ 0)

Equation (15) may be written as

$$(\mu I - G^{-1}Z)\lambda = 0 \tag{C1}$$

so that the desired solution λ is the eigenvector of $G^{-1}Z$ corresponding to its largest eigenvalue.

Case 2: Noise on Output Only (NSPQ ≠ 0, NSPW = 0)

By partitioning the matrix G into four (n+1) x (n+1) blocks and correspondingly partitioning λ one obtains

$$\left[\begin{bmatrix} G_{11} & G_{12} \\ G_{21} & G_{22} \end{bmatrix} - \beta \begin{bmatrix} I & 0 \\ 0 & 0 \end{bmatrix}\right] \begin{bmatrix} \lambda^{(1)} \\ \lambda^{(2)} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \tag{C2}$$

which is equivalent to solving the pair

$$[(G_{11} - G_{12}G_{22}{}^{-1}G_{21}) - \beta I]\ \lambda^{(1)} = 0 \tag{C3}$$

$$\lambda^{(2)} = -G_{22}{}^{-1}G_{21}\ \lambda^{(1)} \tag{C4}$$

The first part is solved as a usual eigenvalue problem. The eigenvector $\lambda^{(1)}$ corresponding to the minimum eigenvalue is selected, and, then, from the second equation $\lambda^{(2)}$ is obtained. The desired parameter vector is finally obtained as

$$\hat{\lambda} = \frac{1}{\lambda^{(1)}(1)} \begin{bmatrix} \lambda^{(1)} \\ \lambda^{(2)} \end{bmatrix} \tag{C5}$$

Case 3: Noise on Input Only (NSPQ = 0, NSPW ≠ 0)

This case is quite similar in nature to case 2 above and is treated accordingly.

(Reverse Page C2 Blank)